

# 大海捞针：使用沙箱捕获多个零日漏洞

**李琦**

安全开发工程师

**金权**

漏洞挖掘和利用工程师

2019-5-30

# 关于我们



**李琦** (@leeqwind)

**360 核心安全高级威胁自动化团队**  
**安全开发工程师**



**金权** (@jq0904)

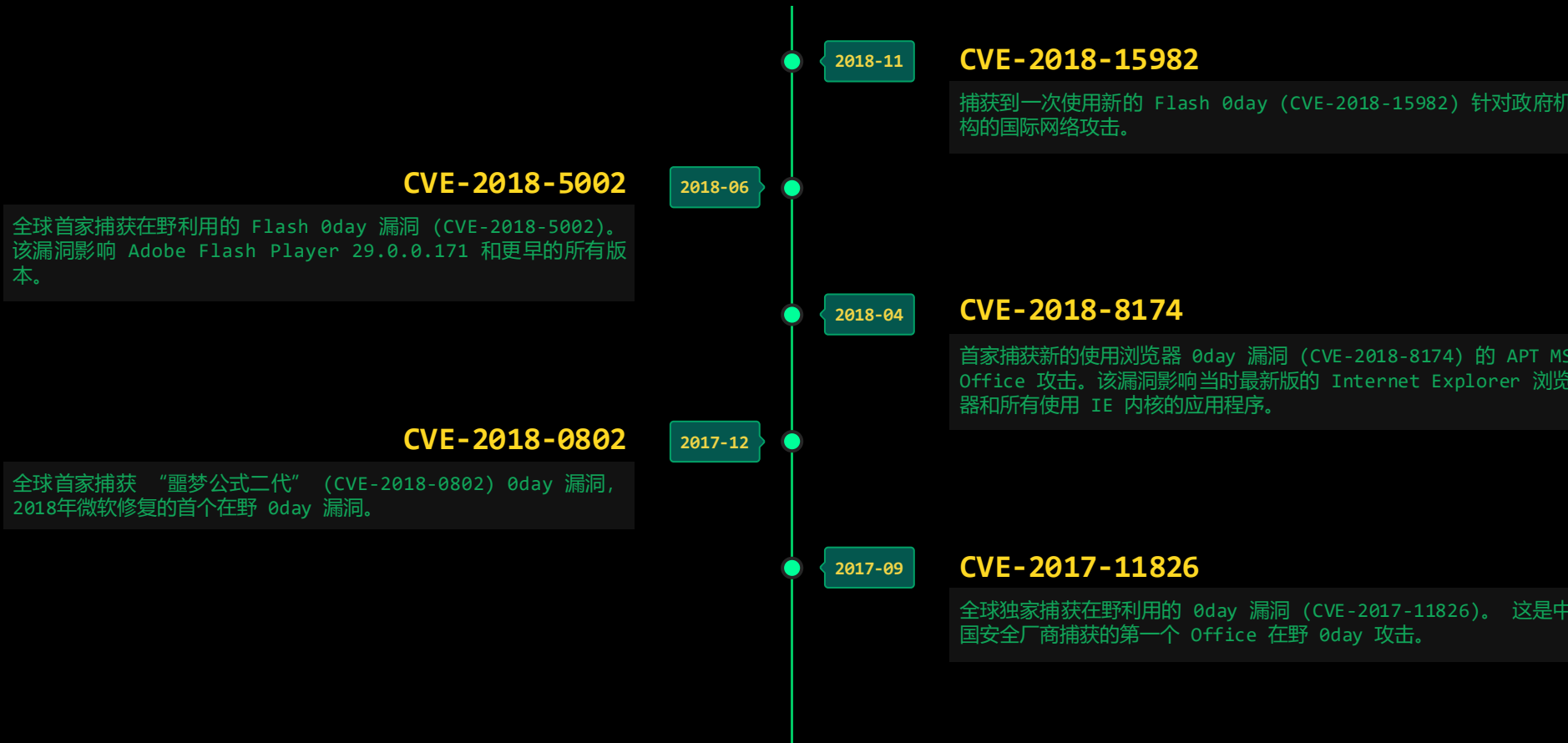
**360 核心安全高级威胁自动化团队**  
**漏洞挖掘和利用工程师**

# 大纲



- 高级威胁自动化和沙箱
- 使用沙箱发现在野 0day 漏洞

# 无处不在的网络攻击

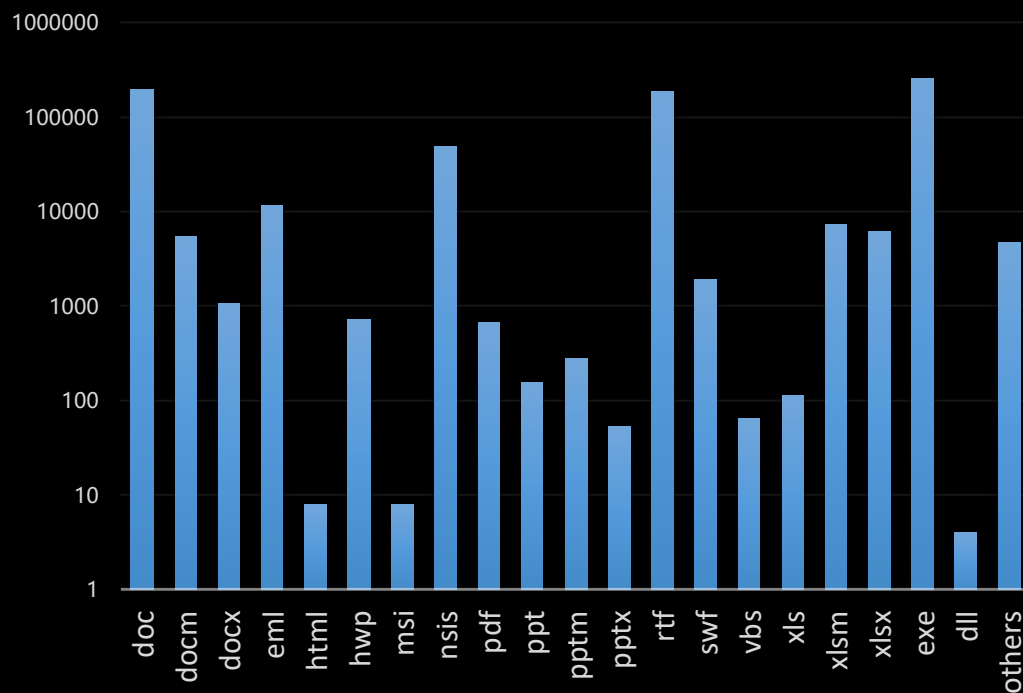


我们至今捕获的五次在野利用 0day 漏洞的攻击事件

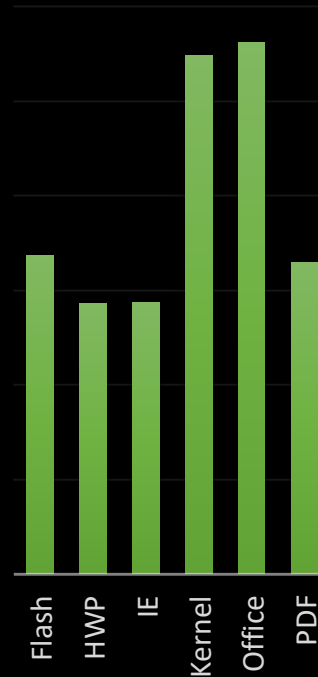
# 无处不在的网络攻击



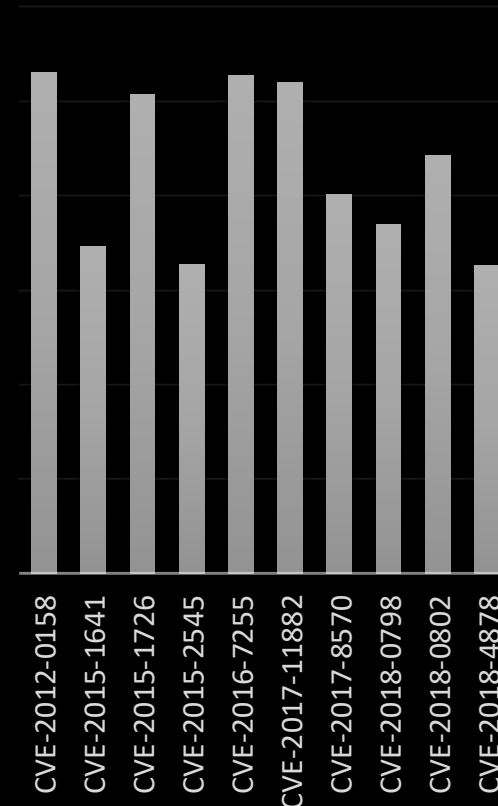
### 部分文件类型分类



### 漏洞模块分类



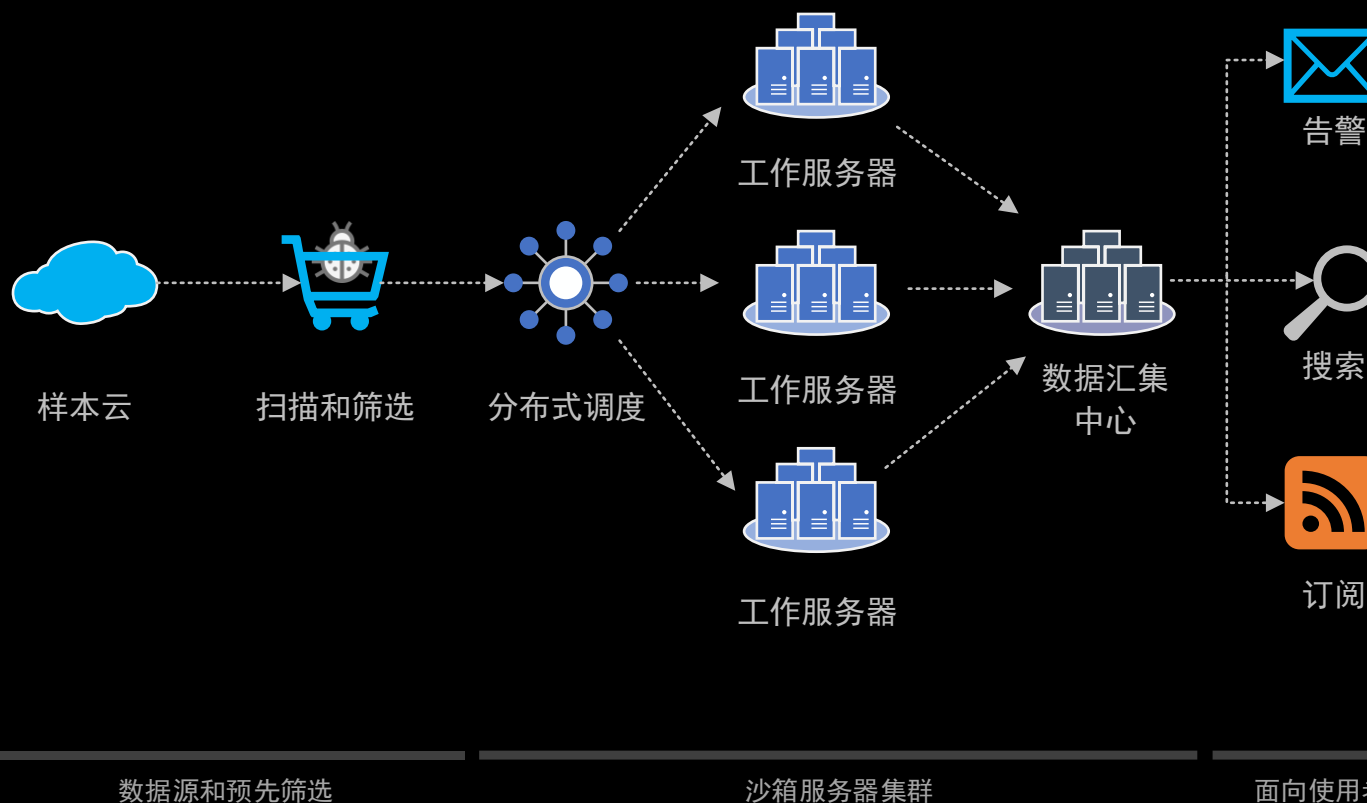
### 部分漏洞计数



从 2018-03 至 2019-03 期间我们监测到的部分 N-day 漏洞攻击文件统计

# 高级威胁自动化

- 大规模的样本云
- 静态反病毒引擎
  - AVE QEX QVM
- 样本预筛选策略
- 沙箱服务器集群
  - 虚拟机隔离环境
  - 沙箱自动化检测引擎
  - 规则评分系统
- 检测结果告警和响应



# 高级威胁自动化

- 大规模的样本云
- 静态反病毒引擎
  - AVE QEX QVM
- 样本预筛选策略
- 沙箱服务器集群
  - 虚拟机隔离环境
  - 沙箱自动化检测引擎
  - 规则评分系统
- 检测结果告警和响应

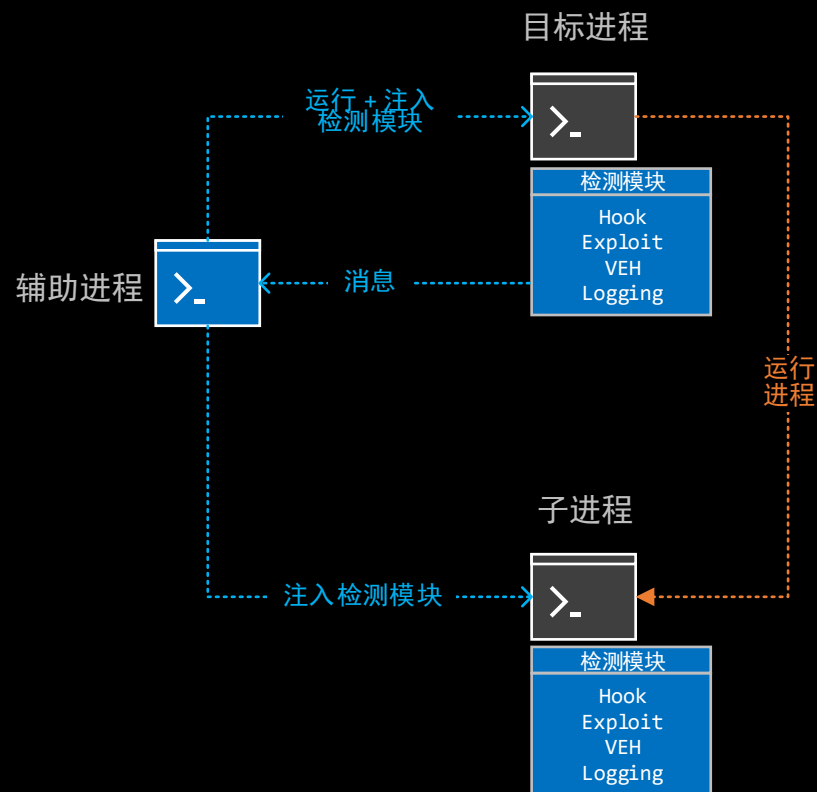


沙箱检测引擎

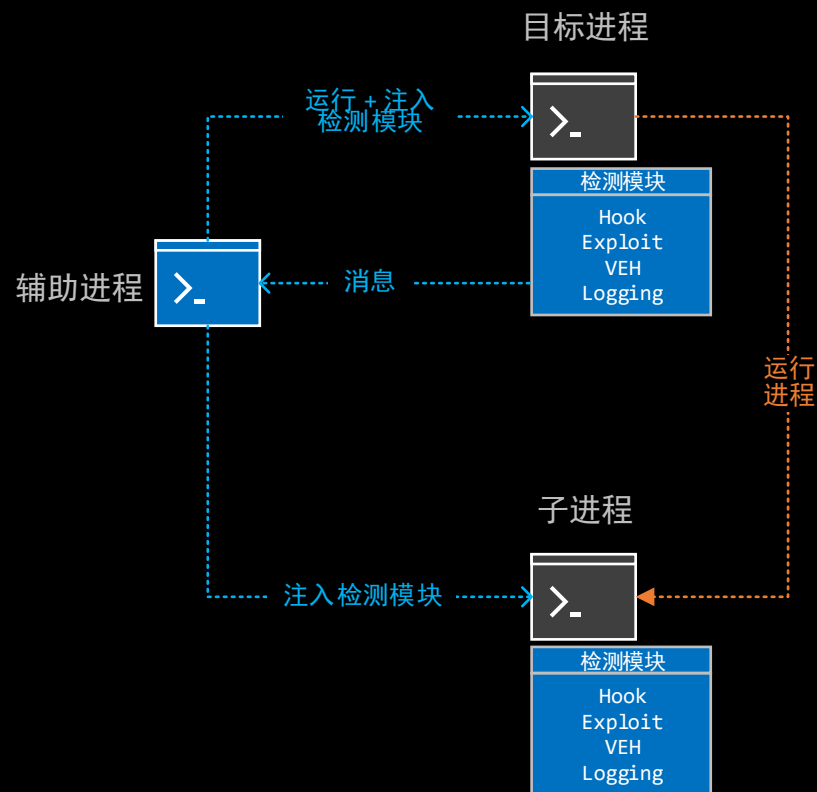
该怎么做？



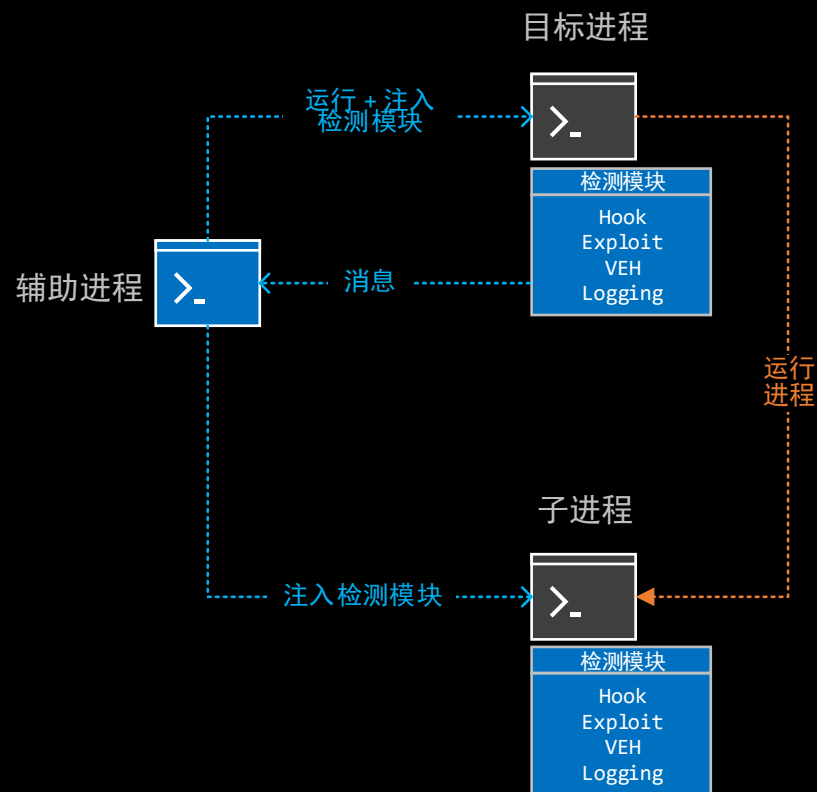
- 最初的方案：动态库
- 注入目标进程执行检测功能
- 挂钩各系统动态库导出函数



- 最初的方案：动态库
- 注入目标进程执行检测功能
- 挂钩各系统动态库导出函数
- 轻量级 😊



- 最初的方案：动态库
- 注入目标进程执行检测功能
- 挂钩各系统动态库导出函数
- 轻量级 😊
- 这样就可以了吗?



# 沙箱检测引擎

- 最初的方案：动态库
- 注入目标进程执行检测功能
- 挂钩各系统动态库导出函数
- 轻量级 😊
- 这样就可以了吗?
- 容易被探测 😞



- 最初的方案：动态库
- 注入目标进程执行检测功能
- 挂钩各系统动态库导出函数
- 轻量级 😊
- 这样就可以了吗?
- 容易被探测 😞
- 容易被绕过 😞

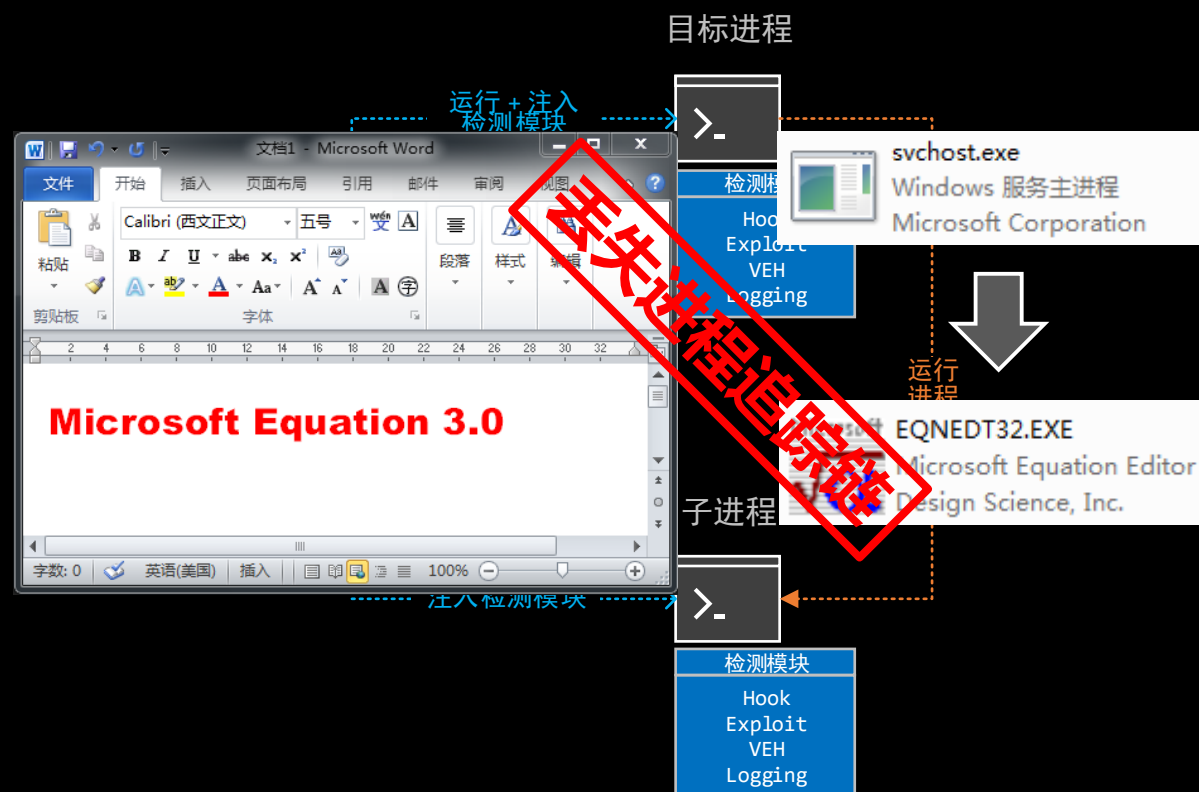
```
00002960          sub_2960      proc near
00002960 49 89 CA      mov     r10, rcx
00002963 B8 7F 10 00 00  mov     eax, 107Fh
00002968 0F 05        syscall
0000296A C3           retn
0000296A          sub_2960      endp
```

```
0:007> uf 077fd000
077fd000 81ec00080000 sub esp,800h
077fd006 60 pushad
[ ... ]
077fd01d ba1d5fc177 mov edx,offset ntdll!NtPrivilegedServiceAuditAlarm+0x5 (77ca5edd)
077fd022 8d45fc lea eax,[ebp-4]
077fd025 50 push eax
[ ... ]
077fd032 b8d7000000 mov eax,0D7h
077fd037 ffd2 call edx
[ ... ]
0:007> uf ntdll!NtPrivilegedServiceAuditAlarm
ntdll!NtPrivilegedServiceAuditAlarm:
77ca5ed8 b8d3000000 mov eax,0D3h
77ca5edd ba0003fe7f mov edx,offset SharedUserData!SystemCallStub (7ffe0300)
77ca5ee2 ff12 call dword ptr [edx]
77ca5ee4 c21400 ret 14h
0:000> uf ntdll!NtProtectVirtualMemory
ntdll!NtProtectVirtualMemory:
77ca5f18 b8d7000000 mov eax,0D7h
77ca5f1d ba0003fe7f mov edx,offset SharedUserData!SystemCallStub (7ffe0300)
77ca5f22 ff12 call dword ptr [edx]
77ca5f24 c21400 ret 14h
```

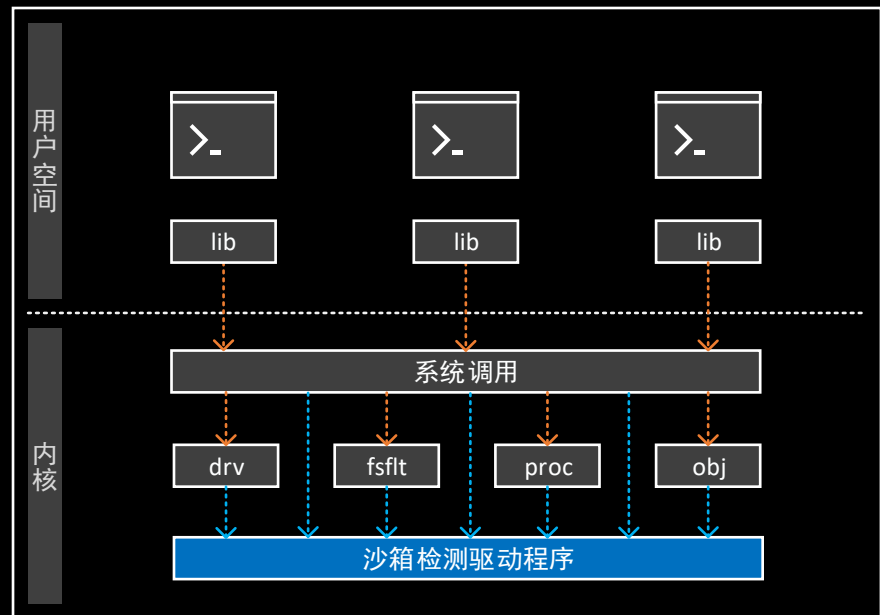
绕过用户态检测

# 沙箱检测引擎

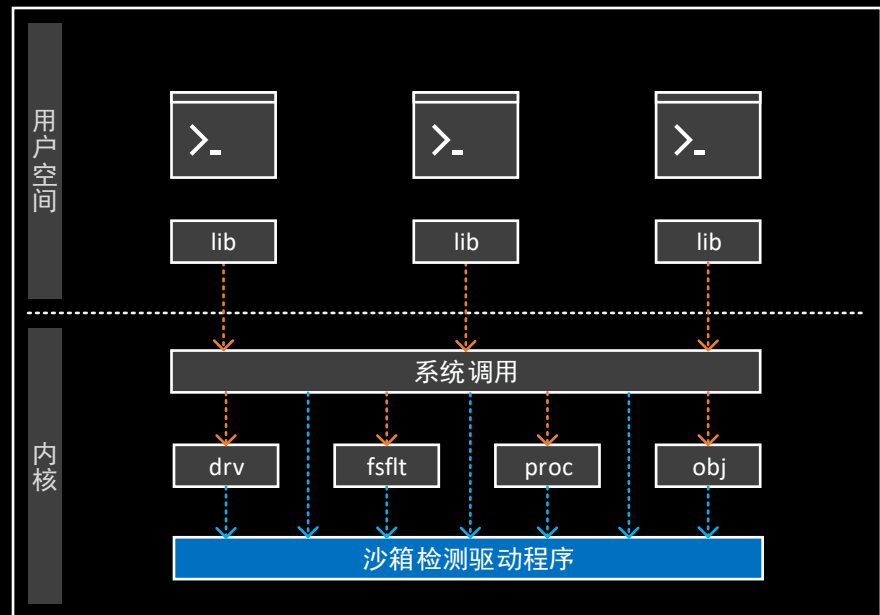
- 最初的方案：动态库
- 注入目标进程执行检测功能
- 挂钩各系统动态库导出函数
- 轻量级 😊
- 这样就可以了吗?
- 容易被探测 😞
- 容易被绕过 😞
- 容易丢失远程方式启动进程的追踪链 😞



- 第二种方案：驱动程序
- 内核层监控目标系统调用
- 系统回调，通知，过滤

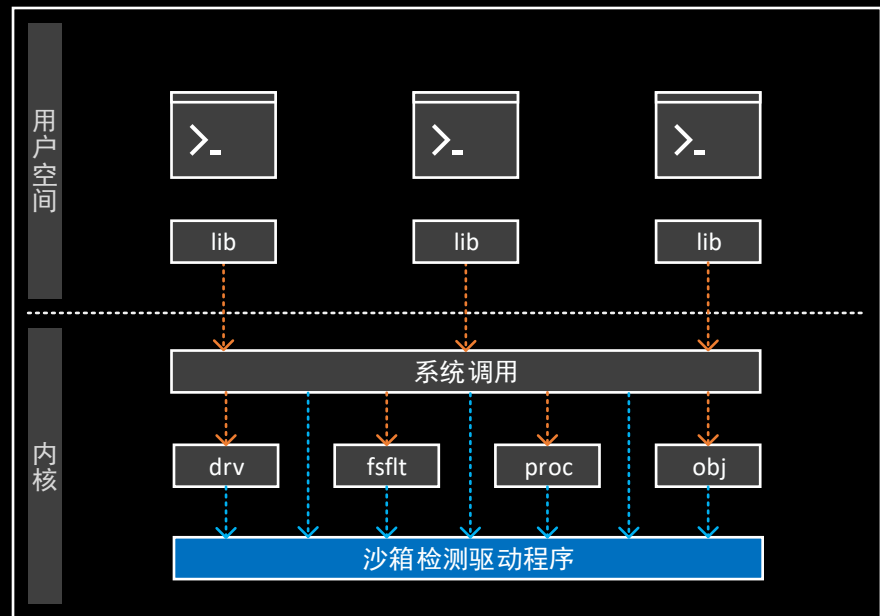


- 第二种方案：驱动程序
- 内核层监控目标系统调用
- 系统回调，通知，过滤
- 更完整监控覆盖面 😊
- 更全面的污点追踪 😊





- 第二种方案：驱动程序
- 内核层监控目标系统调用
- 系统回调，通知，过滤
- 更完整监控覆盖面 😊
- 更全面的污点追踪 😊
- 这就没问题了吗？



- 第二种方案：驱动程序
- 内核层监控目标系统调用
- 系统回调，通知，过滤
- 更完整监控覆盖面 😊
- 更全面的污点追踪 😊
- 这就没问题了吗?
- 64 位操作系统的 PATCH GUARD 机制 😞

```
A problem has been detected and windows has been shut down to prevent damage
to your computer.

Modification of system code or a critical data structure was detected.

If this is the first time you've seen this Stop error screen,
restart your computer. If this screen appears again, follow
these steps:

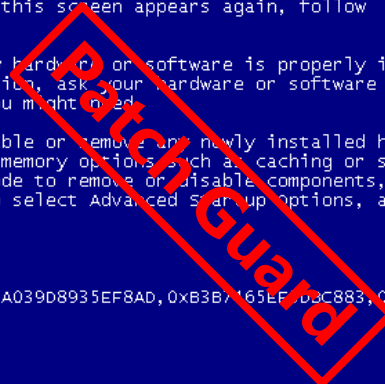
Check to make sure any new hardware or software is properly installed.
If this is a new installation, ask your hardware or software manufacturer
for any windows updates you might need.

If problems continue, disable or remove any newly installed hardware
or software. Disable BIOS memory options such as caching or shadowing.
If you need to use safe mode to remove or disable components, restart
your computer, press F8 to select Advanced Start up options, and then
select safe mode.

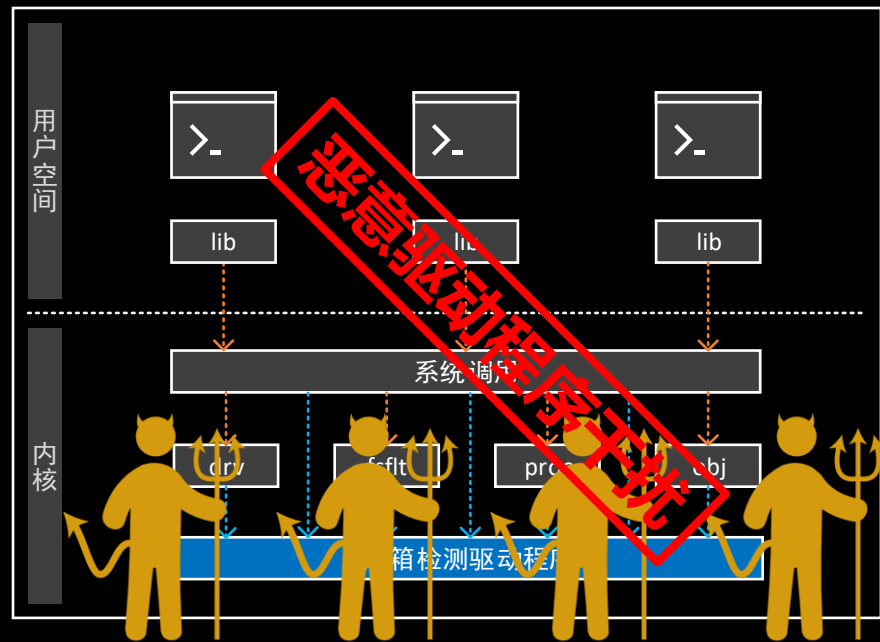
Technical information:

*** STOP: 0x00000109 (0xA3A039D8935EF8AD,0xB3B7165EF,0DC883,0xFFFFF80003E704D0,0
x00000000000000001)

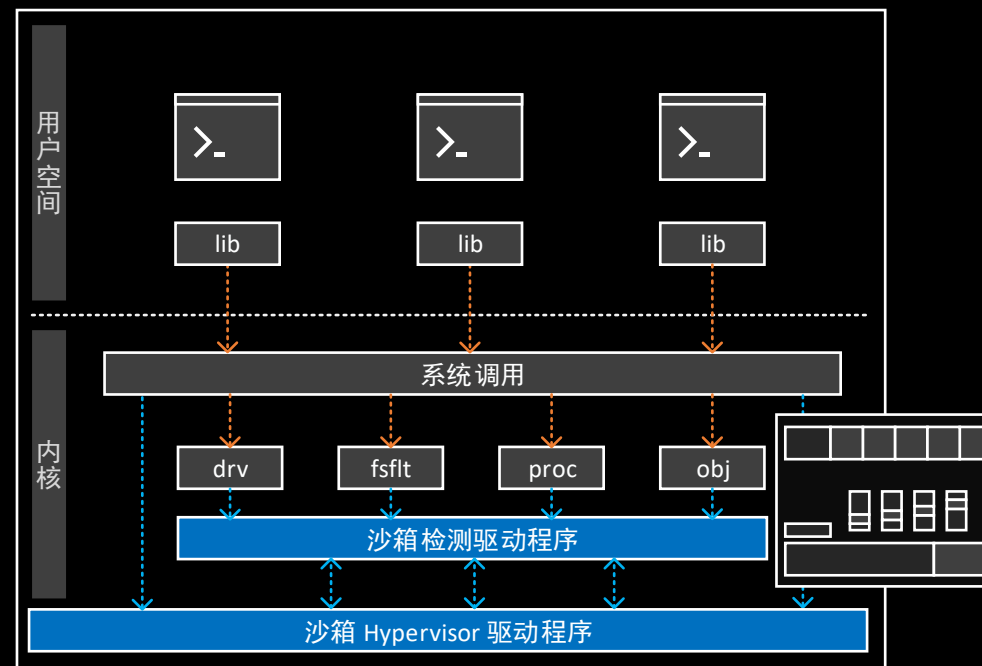
Collecting data for crash dump ...
Initializing disk for crash dump ...
```



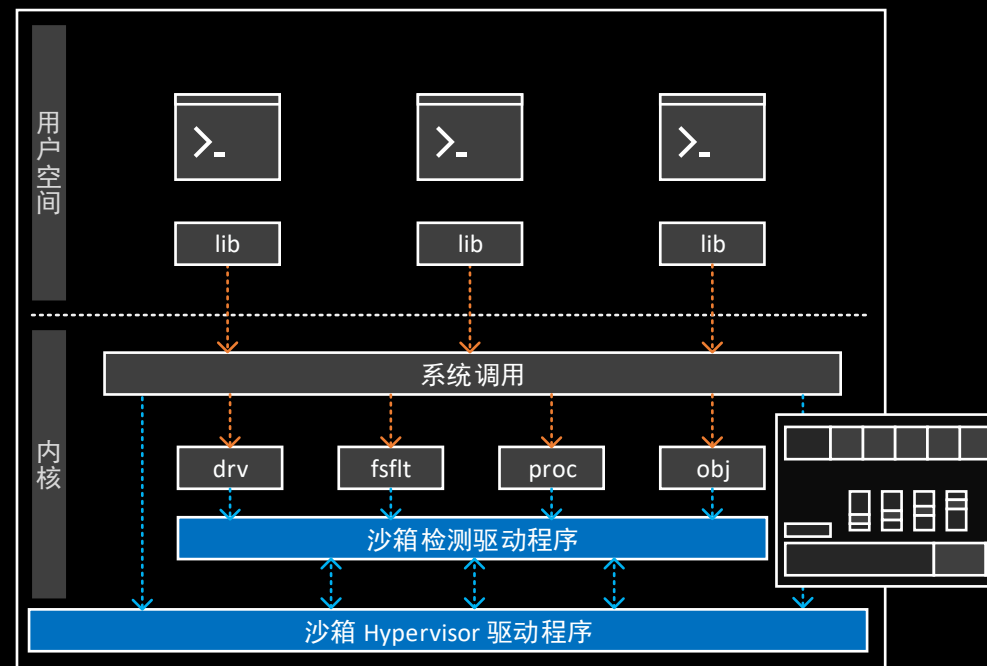
- 第二种方案：驱动程序
- 内核层监控目标系统调用
- 系统回调，通知，过滤
- 更完整监控覆盖面 😊
- 更全面的污点追踪 😊
- 这就没问题了吗?
- 64 位操作系统的 PATCH GUARD 机制 😞
- 加载驱动的恶意程序的干扰 😞



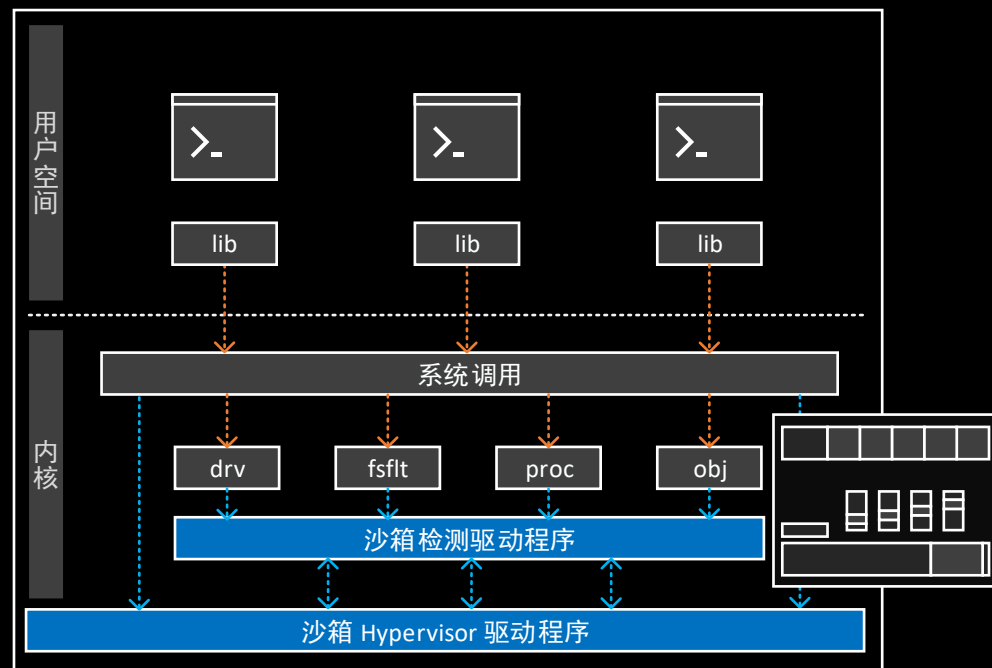
- 第三种方案：基于硬件虚拟化的驱动程序
- 基于虚拟化的系统调用监控功能
- 针对敏感内存读写访问监控功能



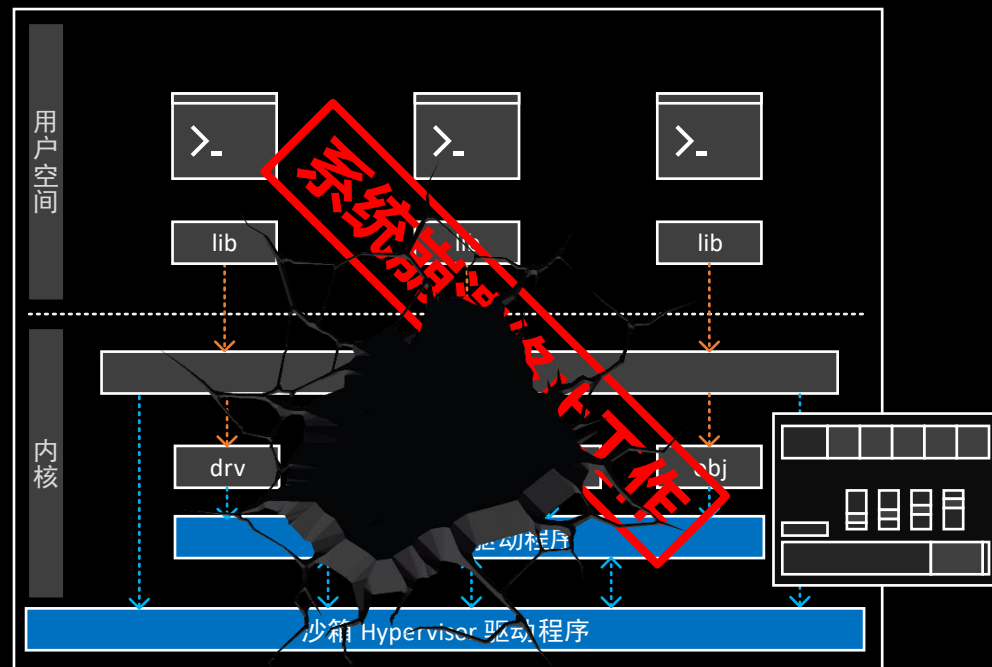
- 第三种方案：基于硬件虚拟化的驱动程序
- 基于虚拟化的系统调用监控功能
- 针对敏感内存读写访问监控功能
- 避免 PATCH GUARD 导致的蓝屏 😊
- 保护自身驱动代码数据 😊
- 拓展更全面的检测功能 😊



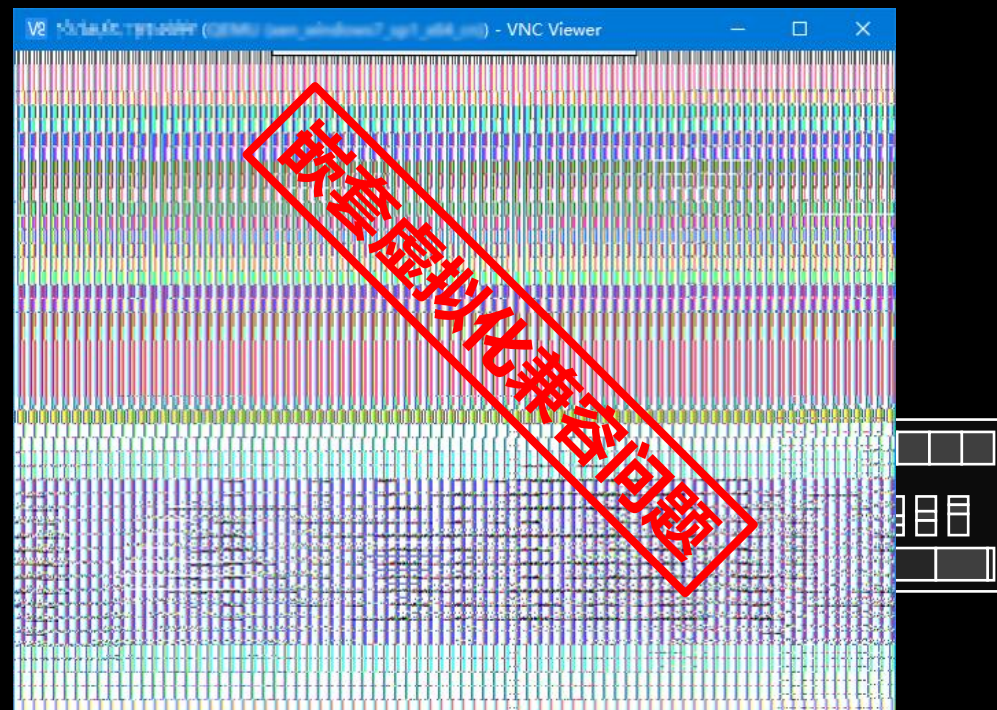
- 第三种方案：基于硬件虚拟化的驱动程序
- 基于虚拟化的系统调用监控功能
- 针对敏感内存读写访问监控功能
- 避免 PATCH GUARD 导致的蓝屏 😊
- 保护自身驱动代码数据 😊
- 拓展更全面的检测功能 😊
- 这样就万无一失了吗?



- 第三种方案：基于硬件虚拟化的驱动程序
- 基于虚拟化的系统调用监控功能
- 针对敏感内存读写访问监控功能
- 避免 PATCH GUARD 导致的蓝屏 😊
- 保护自身驱动代码数据 😊
- 拓展更全面的检测功能 😊
- 这样就万无一失了吗?
- 无法确保其他内核模块的可靠性 😞

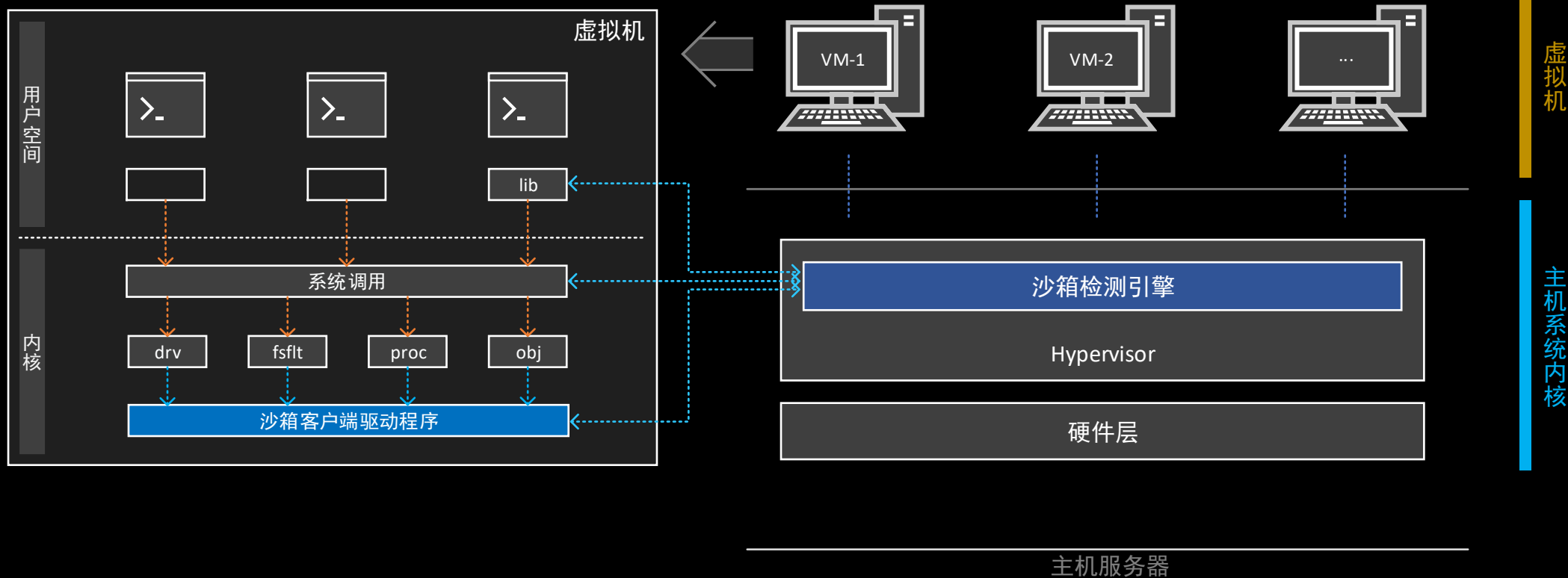


- 第三种方案：基于硬件虚拟化的驱动程序
- 基于虚拟化的系统调用监控功能
- 针对敏感内存读写访问监控功能
- 避免 PATCH GUARD 导致的蓝屏 😊
- 保护自身驱动代码数据 😊
- 拓展更全面的检测功能 😊
- 这样就万无一失了吗?
- 无法确保其他内核模块的可靠性 😞
- 虚拟机软件嵌套虚拟化支持不佳 😞



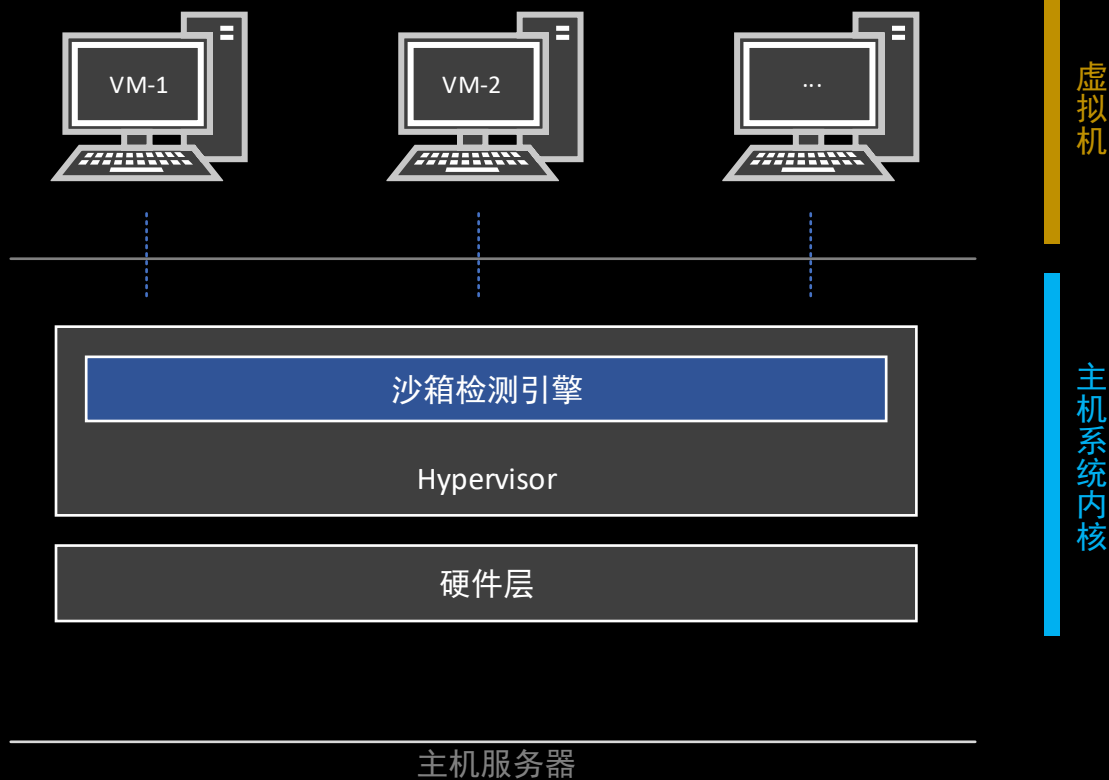


## • 第四种方案：基于全局虚拟机监视器的检测方案



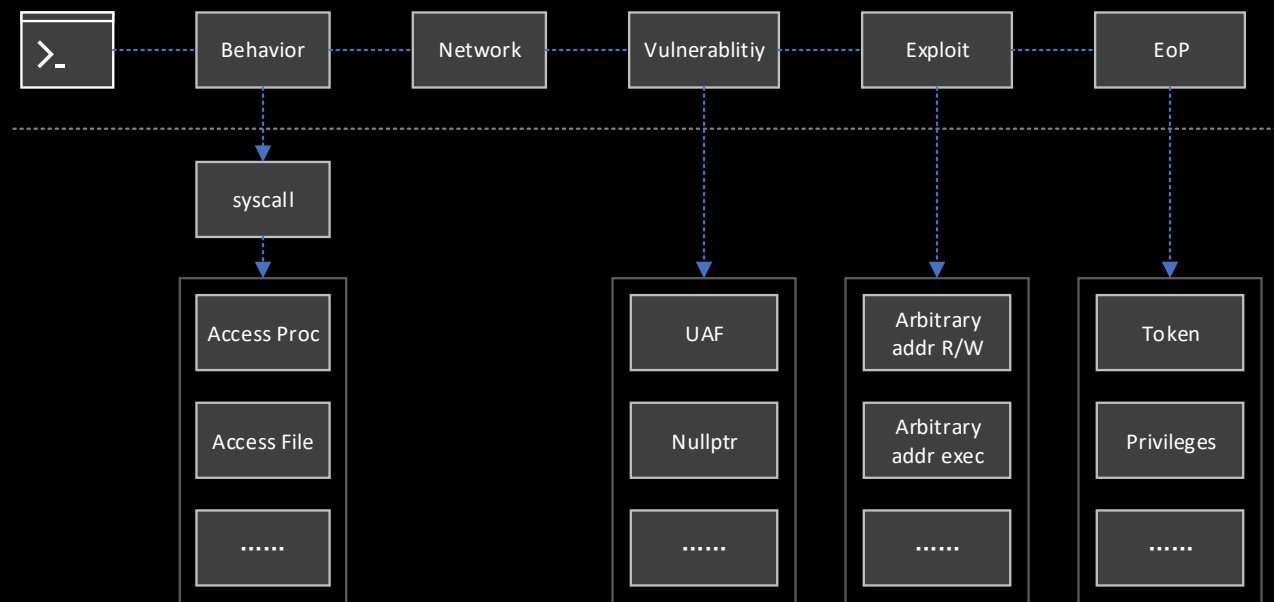
## • 第四种方案：基于全局虚拟机监视器的检测方案

- 核心检测代码位于主机系统内核
- 融合先前各检测方案的优势
- 不依赖虚拟机内部其他系统模块 😊
- 虚拟机崩溃不影响关键检测功能 😊
- 检测数据直接输出主机记录服务 😊



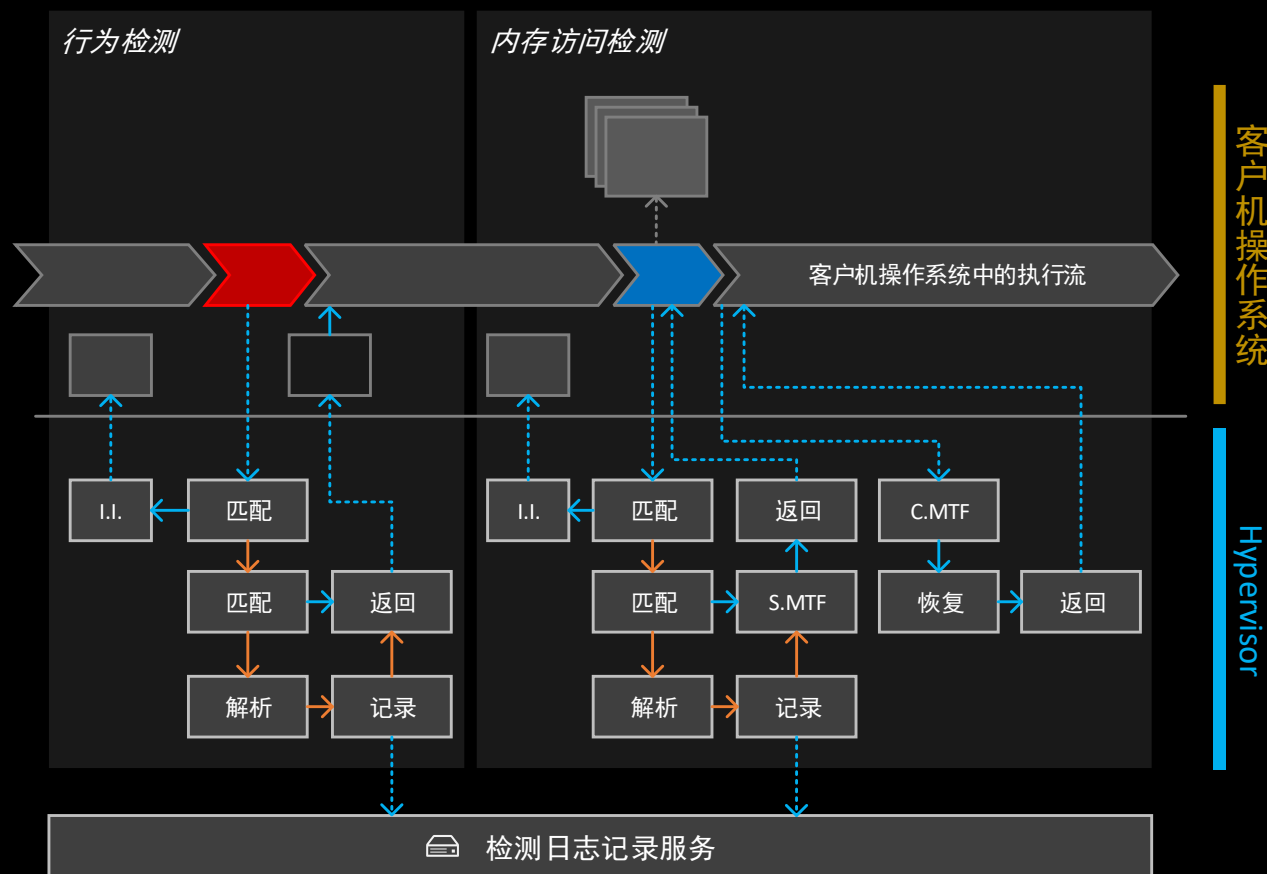
# 沙箱检测技术

- 行为检测
- 内存访问检测
- 内核利用检测
- 内核异常检测
- 已知漏洞检测
- 用户态利用检测



# 沙箱检测技术

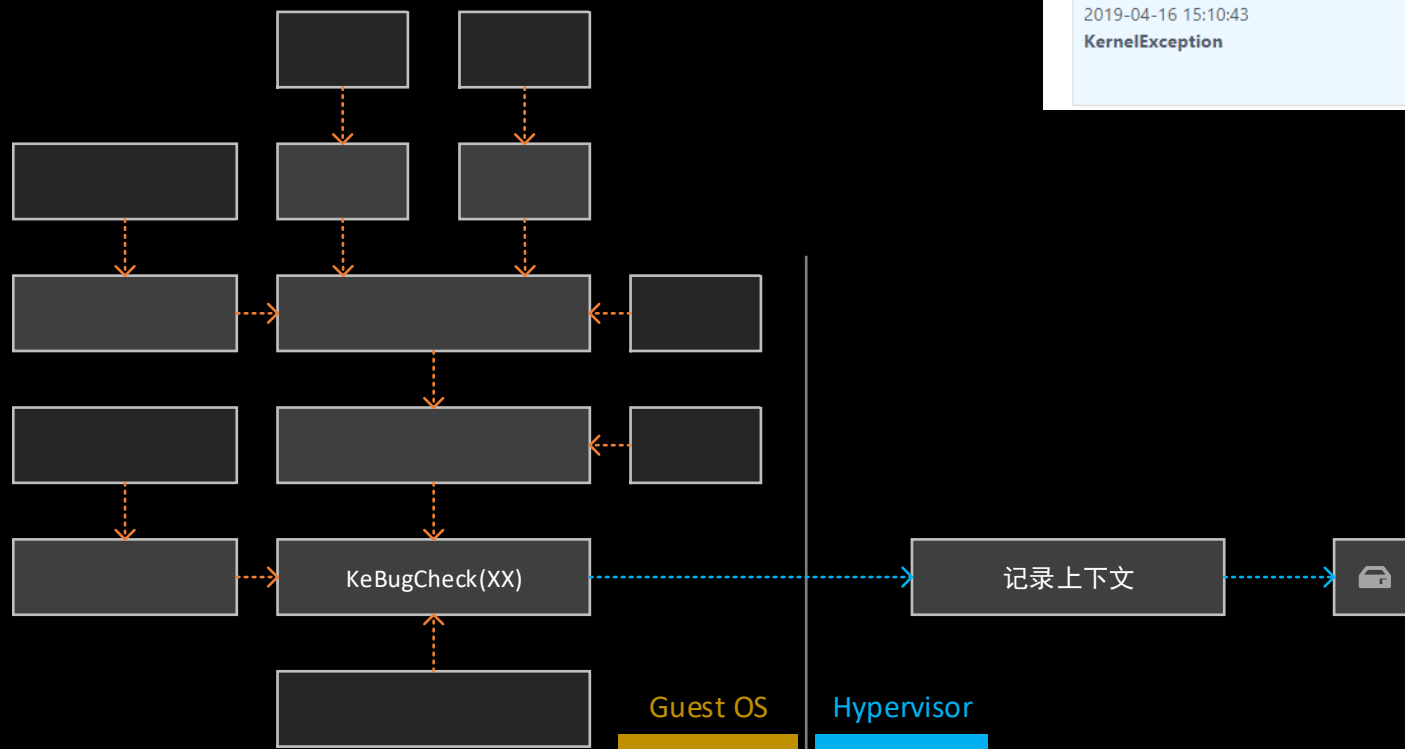
- 行为检测
- 内存访问检测
- 内核利用检测
- 内核异常检测
- 已知漏洞检测
- 用户态利用检测



# 内核利用检测

Vulnerability Triggering	Exploiting	Exploit Result
UAF	Pool/heap spray	Token
Nullptr	Corrupting window	Privileges
OOB	...	Integrity
...		ACL
		...

- 当系统内核发生崩溃时记录关键上下文



Vulnerability and Exploit

Detected blue screen of death (BSOD) happened in the system (1 event)

Event	Context
2019-04-16 15:10:43 KernelException	bugcheck_code: 0xc2 parameter_1: 0x7 parameter_2: 0x1097 parameter_3: 0x5a080063 parameter_4: 0xfe793320

- 精确识别使用已知漏洞进行利用的任务

Vulnerability and Exploit

Detected process hit a kernel hotpatch (1 event)

Event	Context
2019-04-16 06:28:04 <b>KernelHotPatch</b>	checksum: 0x2486d5 module_base: 0x8f8b0000 cve_id: CVE-2018-0817 module_path: C:\WINDOWS\SYSTEM32\WIN32K.SYS

Vulnerability and Exploit

Detected process hit a user hotpatch (2 events)

Event	Context
2019-04-23 19:34:36 <b>UserHotPatch</b>	checksum: 0x85009 module_base: 0x400000 cve_id: CVE-2018-0802 module_path: C:\Program Files\Common Files\Microsoft Shared\EQUATION\EQNEDT32.EXE

- 堆喷射阈值检测
- 导出地址表过滤
- 导入地址表过滤
- ROP 检测
- Flash 针对性检测
  - Vector Length 检测
  - ByteArray Length 检测
  - LoadBytes 转储
  - 其他检测功能
- VBScript 针对性检测
- .....







# 如何使用沙箱发现 0day 漏洞?

**从 CVE-2017-0199 说起...**

- **多环境**
  - 各版本 Windows
  - 各版本 Office
  - 各版本 Flash
- **动态执行**
  - 模拟交互
  - 反静态混淆 (特别是 RTF 文档)
- **记录和还原现场**
- **精确**
  - 漏洞和利用的识别
- **自动化**
  - 自动展示进程行为
  - 自动转储文件
  - 自动转储 LoadBytes 加载的利用代码

# 搭建自动化检测系统



- **历史事件研究**
  - 历史 0day/1day 研究
- **数据源**
  - 360 的海量数据
  - 高质量的共享数据源
- **分析系统**
  - 沙箱
- **推送系统**
- **人工确认**
  - 相关漏洞分析员

# 近6年的相关漏洞



2013	2014	2015	2016	2017	2018
CVE-2013-0634 CVE-2013-3906	CVE-2014-1761 CVE-2014-4114 CVE-2014-6352	CVE-2015-1642 CVE-2015-2424 CVE-2015-2545 CVE-2015-5119 CVE-2015-5122	CVE-2016-4117 CVE-2016-7193 CVE-2016-7855	CVE-2017-0199 CVE-2017-0261 CVE-2017-0262 CVE-2017-8570 CVE-2017-8759 CVE-2017-11292 CVE-2017-11826 CVE-2017-11882	CVE-2018-0798 CVE-2018-0802 CVE-2018-4878 CVE-2018-5002 CVE-2018-8174 CVE-2018-8373 CVE-2018-15982

# 历史漏洞归类



RTF 控制字解析问题	Open XML 标签解析问题	ActiveX 控件解析问题	Office 嵌 Flash 0day
CVE-2010-3333 CVE-2014-1761 CVE-2016-7193	CVE-2015-1641 CVE-2017-11826	CVE-2012-0158 CVE-2012-1856 CVE-2015-2424 CVE-2017-11882 CVE-2018-0798 CVE-2018-0802	CVE-2011-0609 CVE-2011-0611 CVE-2013-0634 HackingTeam 泄露代码 CVE-2016-4117 CVE-2016-7855 CVE-2018-4878 CVE-2018-5002 CVE-2018-15982
TIFF 图片解析问题	EPS 文件解析问题	Moniker	其他 Office 逻辑漏洞
CVE-2013-3906	CVE-2015-2545 CVE-2017-0261 CVE-2017-0262	CVE-2017-0199 CVE-2017-8570 CVE-2017-8759 CVE-2018-8174 CVE-2018-8373	CVE-2014-4114 CVE-2014-6352 CVE-2015-0097



# 历史总是相似的



RTF 控制字解析问题	Open XML 标签解析问题	ActiveX 控件解析问题	Office 嵌 Flash 0day
CVE-2010-3333 CVE-2014-1761 CVE-2016-7193	CVE-2015-1641 <b>CVE-2017-11826</b>	CVE-2012-0158 CVE-2012-1856 CVE-2015-2424 CVE-2017-11882 CVE-2018-0798 <b>CVE-2018-0802</b>	CVE-2011-0609 CVE-2011-0611 CVE-2013-0634 HackingTeam 泄露代码 CVE-2016-4117 CVE-2016-7855 CVE-2018-4878 <b>CVE-2018-5002</b> <b>CVE-2018-15982</b>
TIFF 图片解析问题	EPS 文件解析问题	Moniker	其他 Office 逻辑漏洞
CVE-2013-3906	CVE-2015-2545 CVE-2017-0261 CVE-2017-0262	CVE-2017-0199 CVE-2017-8570 CVE-2017-8759 <b>CVE-2018-8174</b> CVE-2018-8373	CVE-2014-4114 CVE-2014-6352 CVE-2015-0097

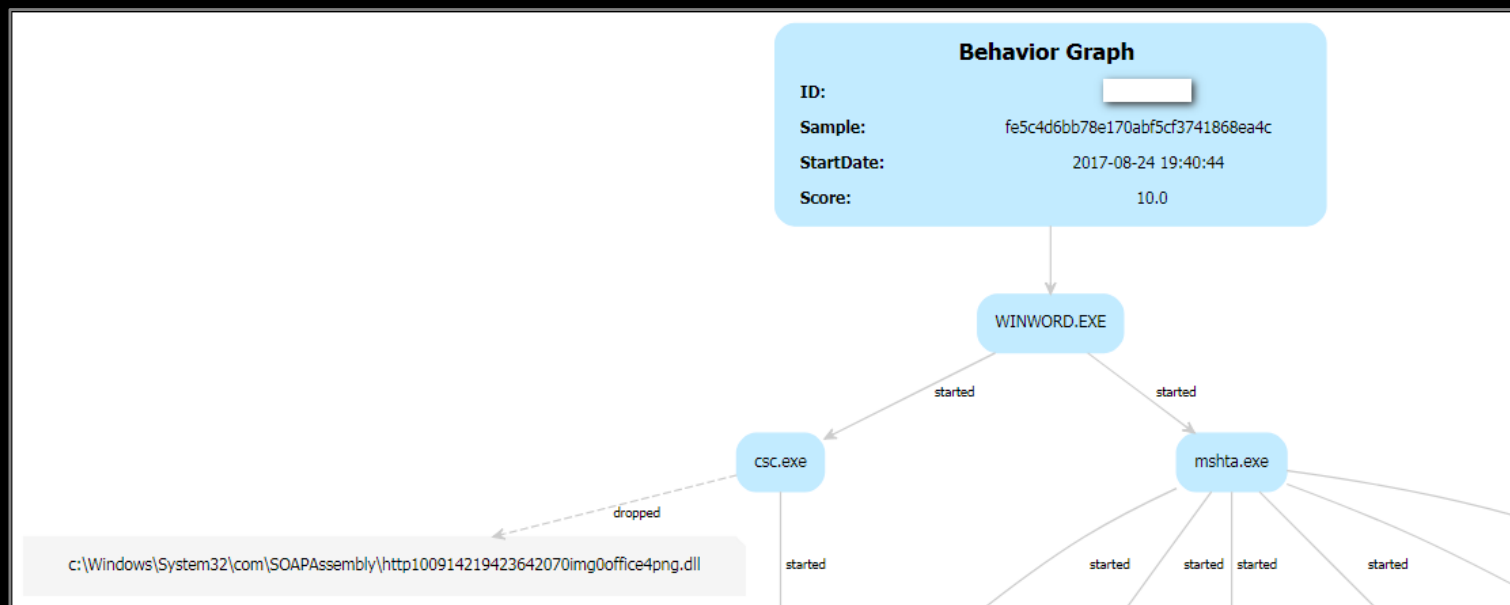
走过的弯路：4个 0day + 1个 1day

- CVE-2017-0261 (0day)
- CVE-2017-0262 (0day) + CVE-2017-0263 (0day)
- 反思
  - 沙箱检测引擎有缺陷 😞
  - CVE-2017-0261 样本无法在 Office 2010 触发 😞
  - CVE-2017-0262 样本无法在 Office 2007 触发 😞
  - 当用户态引擎遇见内核 0day 😞

- CVE-2017-8759 (0day)

- 反思

- 沙箱跑出了样本，但未能及时通知分析人员 😞



- CVE-2017-11292 (1day)

- 反思

- 对 DealersChoice 框架缺乏了解 😞
- 若目标为低版本 Flash, 下发 CVE-2015-7645 😞
- 若目标为高版本 Flash, 下发 CVE-2017-11292 😊

- **DealersChoice**
  - 由 @Unit42\_Intel 命名
  - 被 APT28 使用
  - 持续改进以尽可能地躲避检测
- **初始手法**
  - 检查当前 Flash 版本
  - 地理位置判断
  - 存活时间短
- **新手法**
  - 反沙箱：需要模拟文档下滑
  - 改写开源代码，加入恶意功能，躲避静态检测

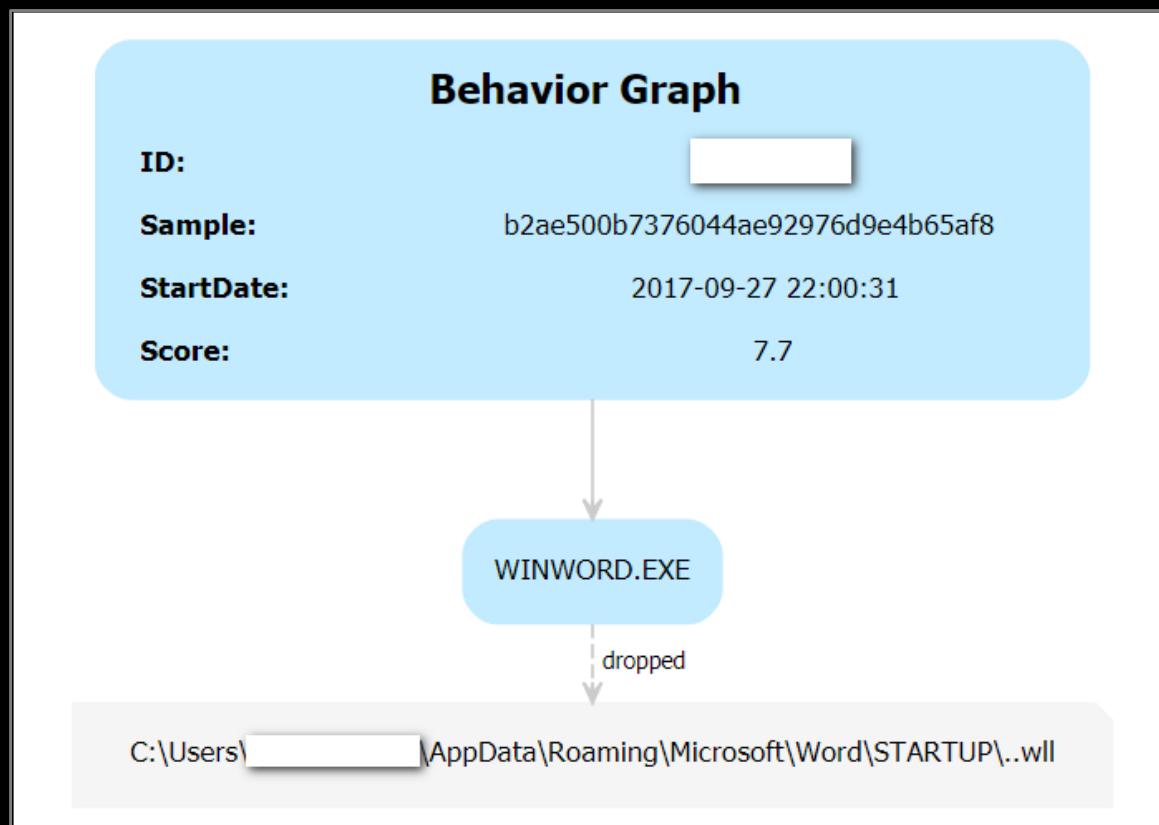
- **沙箱检测引擎缺陷** 😞
  - 开发下一代沙箱检测引擎 😊
- **环境选择不正确** 😞
  - 制作多种环境 😊
  - 制定触发率高的投递策略 😊
- **不能及时通知分析人员** 😞
  - 构建实时推送系统 😊
- **对攻击框架不够熟悉** 😞
  - 研究 DealersChoice 框架 😊
  - 强化 Flash 针对性检测 😊

**CVE-2017-11826**



# 从0到1

• 2017年9月27日



# 从0到1

- 第一次有中国厂商抓到 Office 在野 0day



**Haifei Li** @HaifeiLi · 10 Oct 2017

Wow Qihoo360 detects an Office 0day attack ITW, a fair the 1st time for Chinese security companies? [twitter.com/360CoreSec/sta...](https://twitter.com/360CoreSec/status/918111111)

**360 Core Security** @360CoreSec

Qihoo 360 Core Security detected an in-the-wild attack that leveraged CVE-2017-11826, an office 0day vulnerability..goo.gl/VDmhF5

3 24 33

## • OLEObject & Font 对象类型混淆 + ActiveX 控件堆喷射

; Office 2007 下正常执行时

```
; mov     eax, [eax+44h]
0:000> dc 38450f4 l4c/4
038450f4 0000ffff 0000ffff 00000004 00000004 .....
03845104 00000001 00000000 00000000 00000000 .....
03845114 00000000 ffffffff ffffffff 00000000 .....
03845124 00000000 ffffffff 00000000 00000000 .....
03845134 00000000 01d9ffa0 67a02e58 .....X..g
```

```
; mov     eax, [eax+44h]
0:000> dc 01d9ffa0 l4c/4
01d9ffa0 00000001 00000001 01f47928 00000009 .....(y.....
01d9ffb0 00000000 00000000 00000000 00000000 .....
01d9ffc0 00000000 000004b0 00000000 00000000 .....
01d9ffd0 0005003c 00000000 00000000 00000000 <.....
01d9ffe0 00000002 01f7e0a0 00000000 .....
```

```
; mov     ecx, [eax]
0:000> dd 01f7e0a0 l1
01f7e0a0 65d9420c
```

```
; call    dword ptr [ecx+4]
0:000> dds 65d9420c l2
65d9420c 65b527ad mso!Ordinal1072+0x2dd
65d94210 658bbe71 mso!Ordinal1836+0xaf // AddRef
```

; Office 2007 下触发漏洞时

```
; mov     eax, [eax+44h]
0:000> dc 5998140 l4c/4
05998140 000001de 000000dd 00000015 00000010 .....
05998150 00000000 00000000 00000000 00000000 .....
05998160 00000000 ffffffff ffffffff 00000000 .....
05998170 00000000 ffffffff 00000000 00000000 .....
05998180 00000000 04131700 67110a89 .....g
```

```
; mov     eax, [eax+44h]
0:000> dc 04131700 l4c/4
04131700 0000045f 00000000 00000000 00000000 _.....
04131710 00000000 00000000 00000000 00000000 .....
04131720 00000000 00000000 0069004c 0063006e .....L.i.n.c.
04131730 00720065 00680043 00720061 00680043 e.r.C.h.a.r.C.h.
04131740 00720061 088888ec 006f0066 a.r.....f.o.
```

```
; mov     ecx, [eax]
0:000> dd 088888ec l1
088888ec 088888ec
```

```
; call    dword ptr [ecx+4]
0:000> dds 088883ec l2
088883ec 72980e2b MSVBVM60!IID_IVbaHost+0x127eb
088883f0 72980e2b MSVBVM60!IID_IVbaHost+0x127eb // Stack Pivot
```

**CVE-2018-0802**

**CVE-2018-8174**

**CVE-2018-5002**

**CVE-2018-15982**

# CVE-2018-0802



- 公式编辑器组件栈溢出

- 2017年12月14日

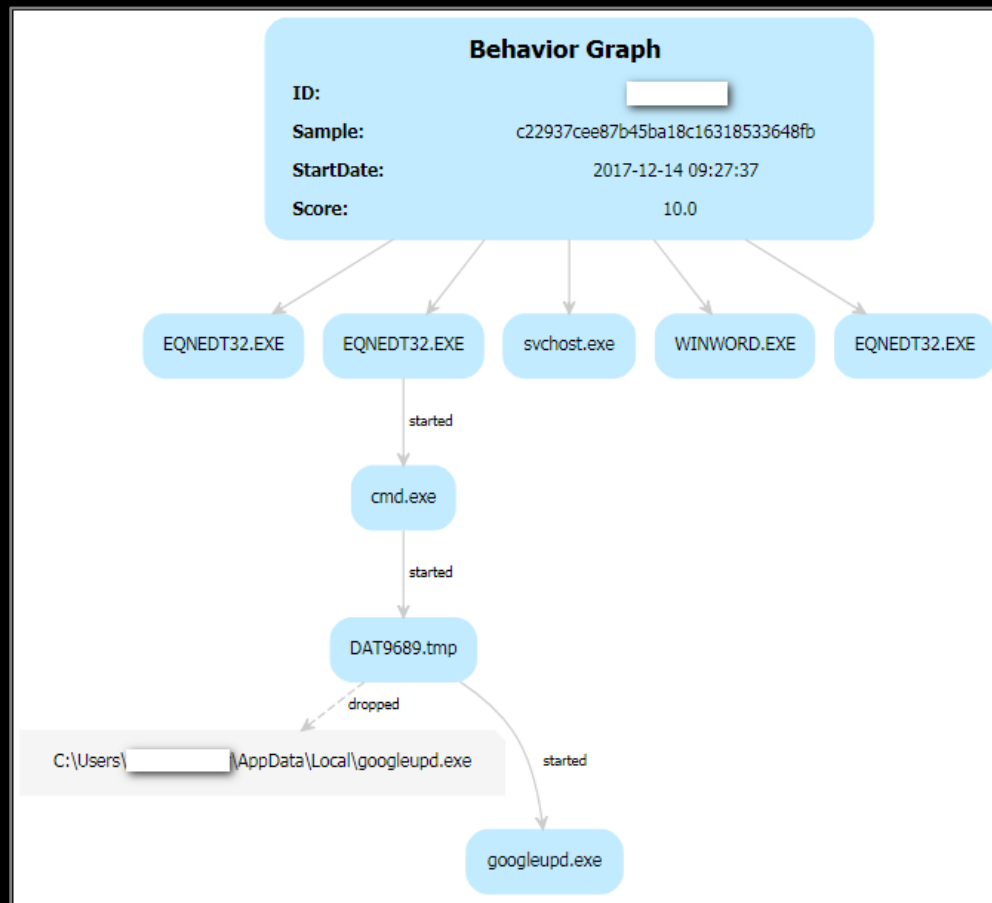
- 同时内嵌两个漏洞

- CVE-2017-11882
- CVE-2018-0802
- 可以正常触发和成功利用 😊

- 2017年12月19日

- 只内嵌一个漏洞

- CVE-2018-0802
- 无法正常触发 😞
- 重新构造 OLE 后可以成功利用 😊



# CVE-2018-0802



- 两个样本都报告给了微软
- 2018年1月10日，微软对我们进行了致谢

## Acknowledgements

Luka Treiber of Opatch Team - ACROS Security

Netanel Ben Simon and Omer Gull of Check Point Software Technologies

Liang Yin of Tencent PC Manager

zhouat of Qihoo 360 Vulcan Team

Zhiyuan Zheng

Yuki Chen of Qihoo 360 Vulcan Team

Yang Kang, Ding Maoyin and Song Shenlei, and Jinquan of Qihoo 360 Core Security (@360CoreSec)

bee13oy of Qihoo 360 Vulcan Team



# CVE-2018-0802



- 哪里错了?
  - 提取解混淆后的 OLE 对象

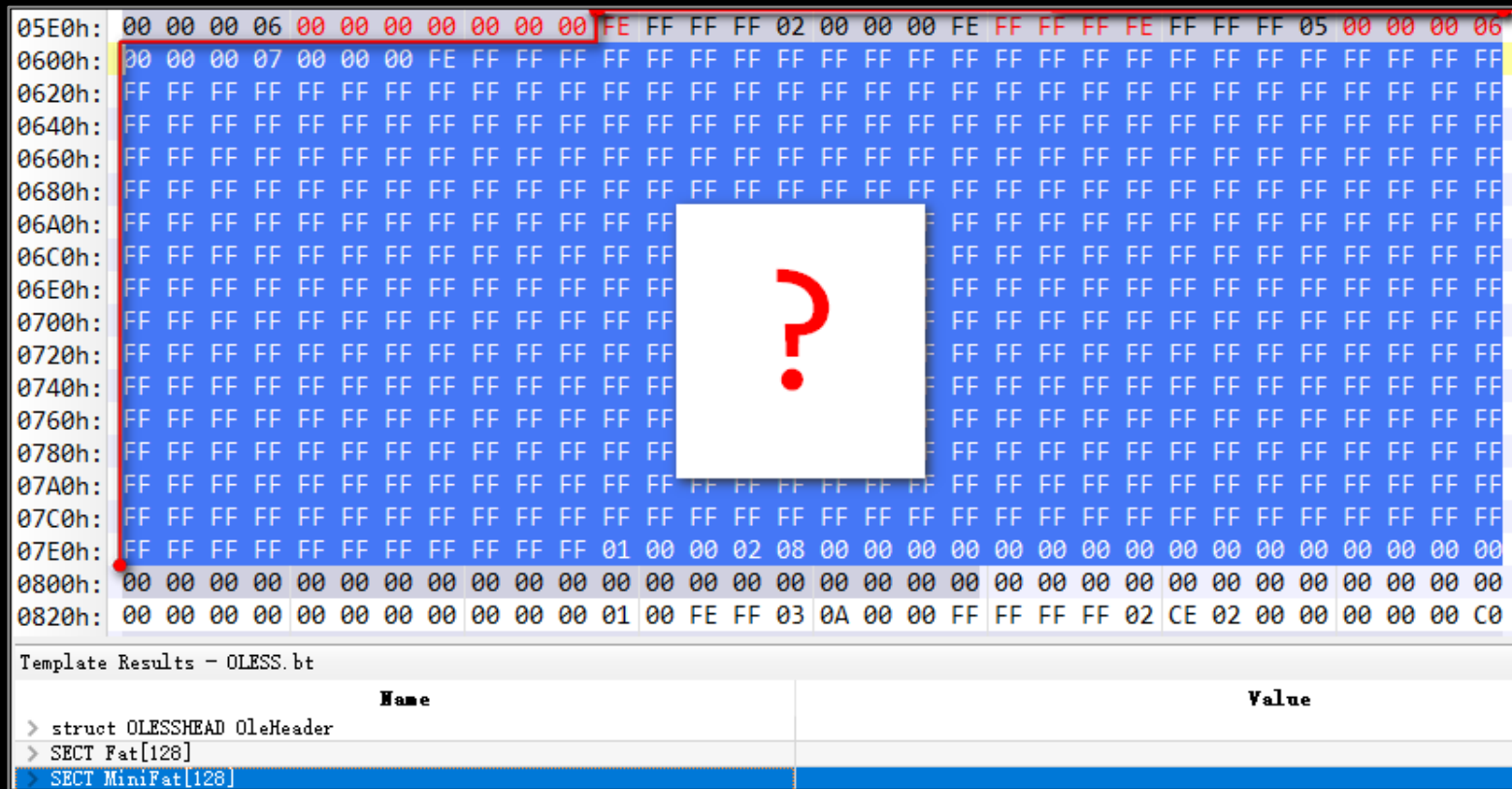
```
0:010> bp ole32!OleConvertOLESTREAMToIStorage
0:010> g
Breakpoint 0 hit
eax=000004e0 ebx=059bc3c0 ecx=00008000 edx=00000000 esi=02d80960 edi=001dade8
eip=75c528fa esp=001dab2c ebp=001dadab iopl=0         nv up ei pl nz na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00200206
ole32!OleConvertOLESTREAMToIStorage:
75c528fa 8bff          mov     edi,edi
0:000> .writemem C:\de-obfuscated_ole.bin poi(poi(poi(esp + 0x04) + 0x08)) Lpoi(poi(esp + 0x04) + 0x0C)
Writing dc5 bytes..

0:000> db poi(poi(poi(esp + 0x04) + 0x08))
04946510  01 05 00 00 02 00 00 00-0b 00 00 00 45 71 75 61  .....Equa
04946520  74 69 6f 6e 2e 33 00 00-00 00 00 00 00 00 00 00  tion.3.....
04946530  0e 00 00 d0 cf 11 e0 a1-b1 1a e1 00 00 00 00 00  .....
04946540  00 00 00 00 00 00 00 00-00 00 00 3e 00 03 00 fe  .....>....
04946550  ff 09 00 06 00 00 00 00-00 00 00 00 00 00 00 01  .....
04946560  00 00 00 01 00 00 00 00-00 00 00 00 10 00 00 02  .....
04946570  00 00 00 01 00 00 00 fe-ff ff ff 00 00 00 00 00  .....
04946580  00 00 00 ff ff ff ff ff-ff ff ff ff ff ff ff  .....
```



# CVE-2018-0802

- 哪里错了?
  - MiniFat Sector 错位 0x15 字节



```
05E0h: 00 00 00 06 00 00 00 00 00 00 00 FE FF FF FF 02 00 00 00 FE FF FF FF FE FF FF FF 05 00 00 00 06
0600h: 00 00 00 07 00 00 00 FE FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0620h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0640h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0660h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0680h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
06A0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
06C0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
06E0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0700h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0720h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0740h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0760h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0780h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
07A0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
07C0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
07E0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0800h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0820h: 00 00 00 00 00 00 00 00 00 00 00 01 00 FE FF 03 0A 00 00 FF FF FF FF 02 CE 02 00 00 00 00 00 C0
```

Template Results - OLESS.bt

Name	Value
> struct OLESSHEAD OleHeader	
> SECT Fat[128]	
> SECT MiniFat[128]	



# CVE-2018-0802



- 2018年元旦后，更多 CVE-2018-0802 样本出现
- 其他研究员注意到了这些样本，但不知道它们利用了 0day 😞



# 如何区分两个漏洞



```
IPersistStorage::Load(406881)
  offset:406a93    call ReadMTEFData(42f8ff)
    offset:42f921    call 43755c
      offset:4375d5    call 43a720
        offset:43a72a    call 43a87a
          offset:43a89b    call 43b418
            ; Font tag parse Logic
            offset:43b44b    call ReadFontName(4164fa)
            offset:43b461    call 4214c6
              offset:4214dd    call LogfontStruct_Overflow(421774)
                offset:4217c3    call 421e39
                  offset:421e5e    rep movsd <- CVE-2018-0802
                offset:4218cb    call 451d50
                offset:4218df    call 4115a7
                  offset:4115d3    call final_overflow(4115d3)
                    offset:411658    rep movsd <- CVE-2017-11882
                    offset:411874    retn
```

Eqnedt32.exe 2000.11.9.0

# 如何区分多个漏洞



- 准确区分三个公式编辑器漏洞

2018-01-24 01:28:57 <b>UserHotPatch</b>	checksum: 0x85009 module_base: 0x400000  cve_id: CVE-2017-11882 module_path: C:\Program Files\Common Files\Microsoft Shared\EQUATION\EQNEDT32.EXE
2018-09-27 12:37:34 <b>UserHotPatch</b>	checksum: 0x85009 module_base: 0x400000  cve_id: CVE-2018-0798 module_path: C:\Program Files\Common Files\Microsoft Shared\EQUATION\EQNEDT32.EXE
2018-03-30 21:30:29 <b>UserHotPatch</b>	checksum: 0x874f7 module_base: 0x8e0000 cve_id: CVE-2018-0802  module_path: C:\Program Files\*\Microsoft Shared\EQUATION\EQNEDT32.EXE

# CVE-2018-8174

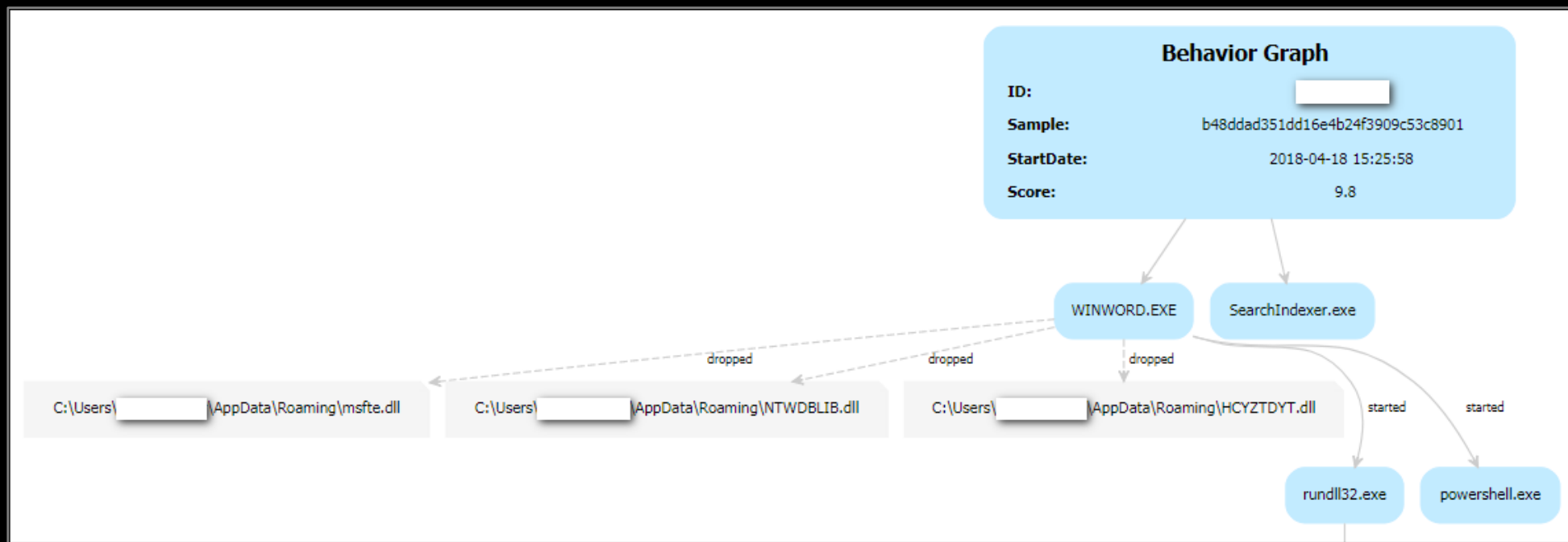


- 两个更早的 Office 样本
- Moniker 远程加载 CVE-2014-6332
  
- **2018年1月17日**
  - Document MD5: A9D3F7A1ACD624DE705CF27EC699B6B6
  - Moniker: hxxp://s.dropcanvas[.]com/1000000/940000/939574/akw.html
  - akw.html MD5: C40A128AE7AEFFA3C1720A516A99BBDF
  
- **2018年2月23日**
  - Document MD5: 2E658D4A286F3A4176A60B2450E9E729
  - Moniker: hxxp://s.dropcanvas[.]com/1000000/942000/941030/IE.html
  - IE.html MD5: C36D544588BAF97838588E732B3D47E9

# CVE-2018-8174



- 2018年4月18日
- RTF 文档加载执行 VBScript 0day



# CVE-2018-8174



- 2018年5月8日, 微软对我们进行了致谢

## Acknowledgements

Anonymous working with Trend Micro's Zero Day Initiative

Vladislav Stolyarov of Kaspersky Lab

Yang Kang of Qihoo 360 Core Security

Ding Maoyin of Qihoo 360 Core Security

Dan Lutas of Bitdefender

Anton Ivanov of Kaspersky Lab

Simon Zuckerbraun working with Trend Micro's Zero Day Initiative

Jinquan of Qihoo 360 Core Security

Song Shenlei of Qihoo 360 Core Security



# CVE-2018-8174



- UAF -> 超长数组 -> 任意地址读写

```
Class class_setprop_a
  Dim mem

  Function P
  End Function

  Function SetProp(Value)
    mem = Value 'callback
    SetProp = 0
  End Function
End Class
```

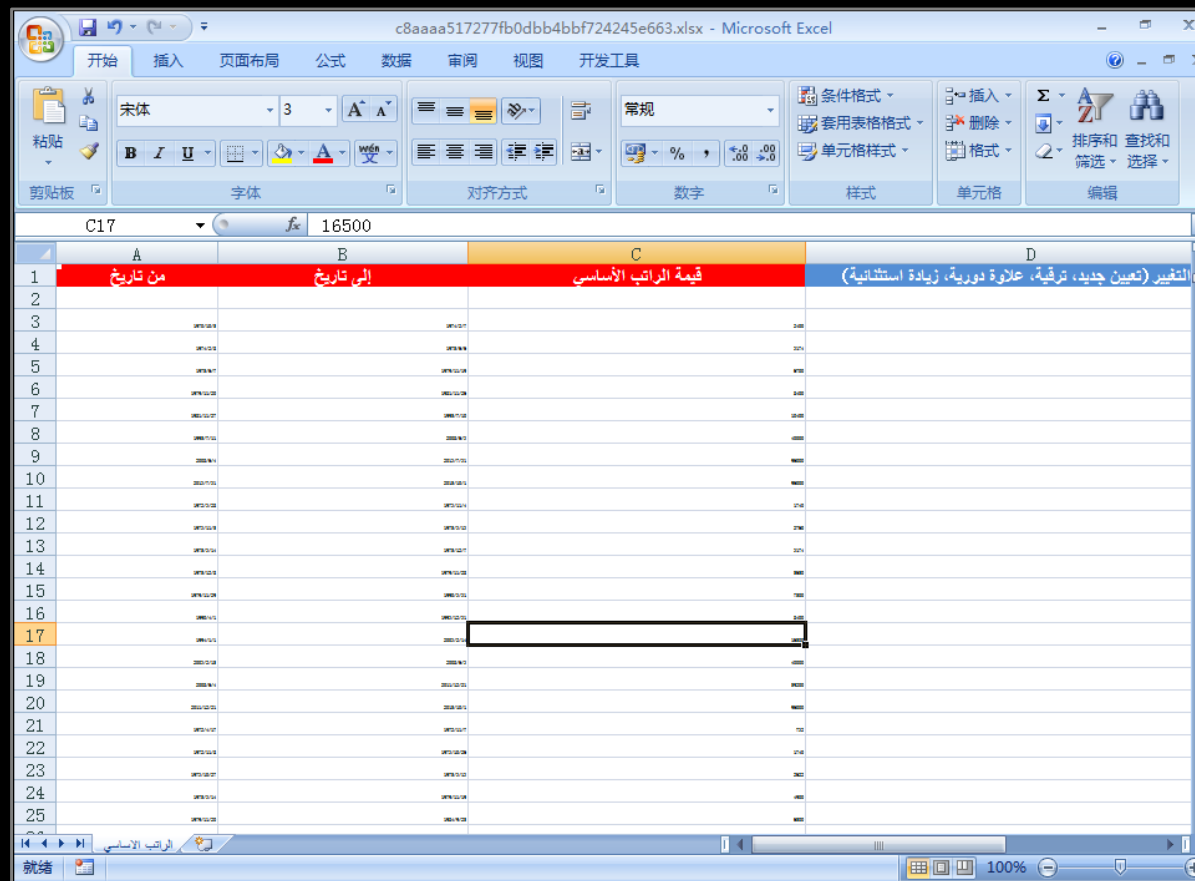
```
// before mem = Value
0:005> dd 022cb91c 14
022cb91c  00000008 00000000 04730834 00000000
0:005> dd 04730834 16
04730834  08800001 00000001 00000000 00000000
04730844  7fffffff 00000000

// after mem = Value
0:007> dd 022cb91c 14
022cb91c  0000200c 00000000 04730834 00000000
0:007> dd 04730834 16
04730834  08800001 00000001 00000000 00000000
04730844  7fffffff 00000000
```

# CVE-2018-5002



- 2018年6月1日
- 一套复杂的 Flash 控制框架
- AVM2 解释器漏洞



# CVE-2018-5002



- 2018年6月7日, Adobe 对我们进行了致谢

• CVE-2018-5002 was independently identified and reported by the following organizations and individuals: Chenming Xu and Jason Jones of ICEBRG, Bai Haowen, Zeng Haitao and Huang Chaowen of 360 Threat Intelligence Center of 360 Enterprise Security Group, and Yang Kang, Hu Jiang, Zhang Qing, and Jin Quan of Qihoo 360 Core Security (@360CoreSec), Tencent PC Manager (<http://guanjia.qq.com/>)

# CVE-2018-5002



- 绕过 ROP 检测 😊
- 覆盖返回地址绕过 CFG 😊
- 无法绕过 EAF 检测 😞

```
var cls25:class_25 = new class_25(cls8, RtlUnwind_Addr);
var NtProtectVirtualMemory_Addr:uint = cls25.GetFuncAddrByEAT("NtProtectVirtualMemory");
if(0 == NtProtectVirtualMemory_Addr)
{
    return new Array();
}

var NtPrivilegedServiceAuditAlarm_Addr:uint = cls25.GetFuncAddrByEAT("NtPrivilegedServiceAuditAlarm");
if(0 == NtPrivilegedServiceAuditAlarm_Addr)
{
    return new Array();
}
```

# 如何调试 CVE-2018-5002

- 逆向 -> ASC2.0 编译 -> 借助 FFDEC 修改字节码 -> 获得可调试的 swf 文件
- 开源的 WinDBG 插件
  - [https://github.com/michaelpdu/flashext\\_pykd](https://github.com/michaelpdu/flashext_pykd)
- 添加3行代码，让插件变得更完美 🤖

```
def callback_after_call_getmethodname(self):
    # dprintln("Enter into callback_after_call_getmethodname")
    reg_eax = reg("eax")
    # dprintln("EAX = " + hex(reg_eax))
    addr_name = ptrPtr(reg_eax + 0x08)
    len_name = ptrPtr(reg_eax + 0x10)

    if 0 == addr_name and 0 != len_name:
        if ptrPtr(reg_eax + 0x0C) != 0:
            addr_name = ptrPtr(ptrPtr(reg_eax + 0x0C) + 0x08)
```

# 调试器中的 CVE-2018-5002

- 触发漏洞 -> 交换指针 -> 类型混淆

// 触发漏洞前

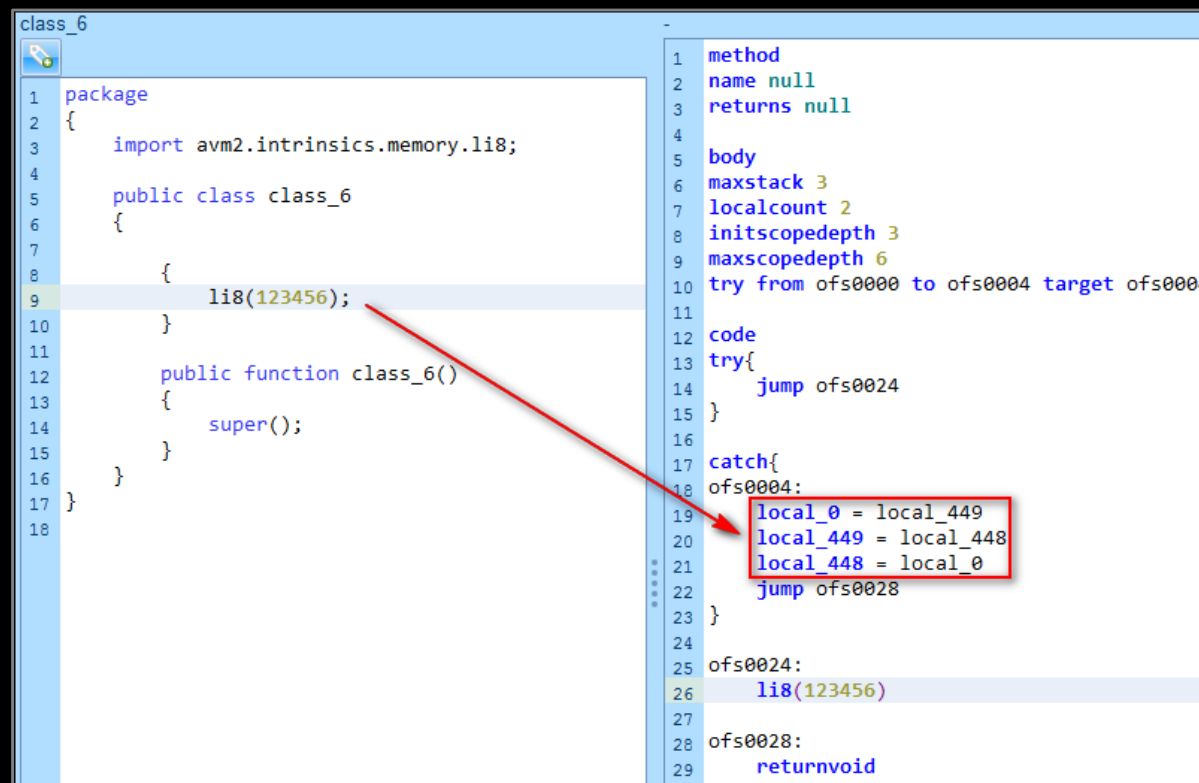
```
0:007> dd 02c0ab24-10
```

```
02c0ab14  093101f0 093101a0 093101f0 093101a0
02c0ab24  093101f0 093101a0 093101f0 093101a0
02c0ab34  093101f0 093101a0 093101f0 093101a0
02c0ab44  093101f0 093101a0 093101f0 093101a0
02c0ab54  093101f0 093101a0 093101f0 093101a0
02c0ab64  093101f0 093101a0 093101f0 093101a0
02c0ab74  093101f0 093101a0 093101f0 093101a0
02c0ab84  093101f0 093101a0 093101f0 093101a0
```

// 触发漏洞后

```
0:007> dd 02c0ab24-10
```

```
02c0ab14  093101f0 093101a0 093101f0 093101f0
02c0ab24  093101a0 093101a0 093101f0 093101a0
02c0ab34  093101f0 093101a0 093101f0 093101a0
02c0ab44  093101f0 093101a0 093101f0 093101a0
02c0ab54  093101f0 093101a0 093101f0 093101a0
02c0ab64  093101f0 093101a0 093101f0 093101a0
02c0ab74  093101f0 093101a0 093101f0 093101a0
02c0ab84  093101f0 093101a0 093101f0 093101a0
```



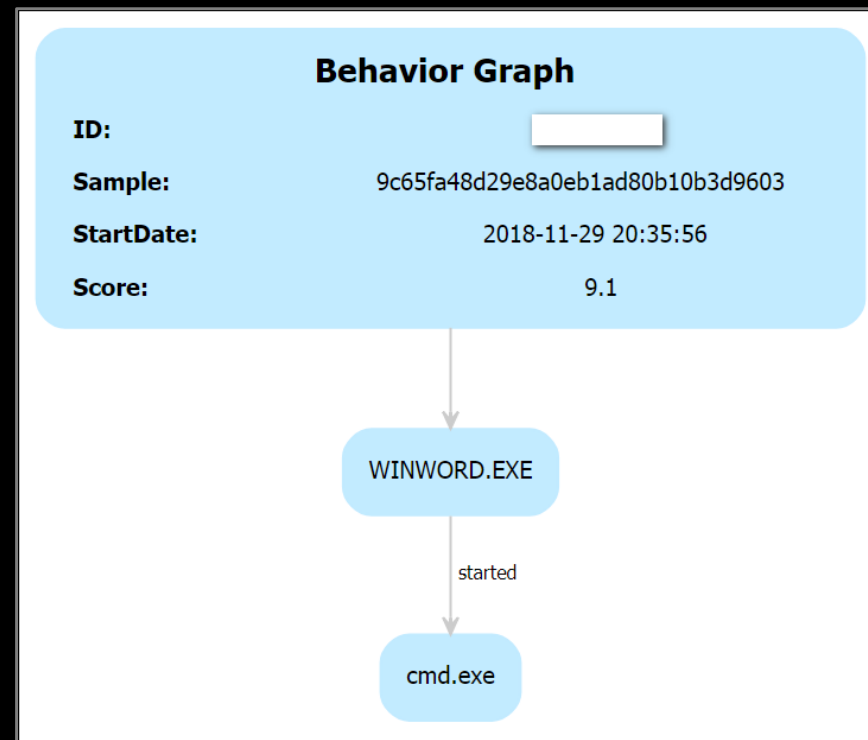
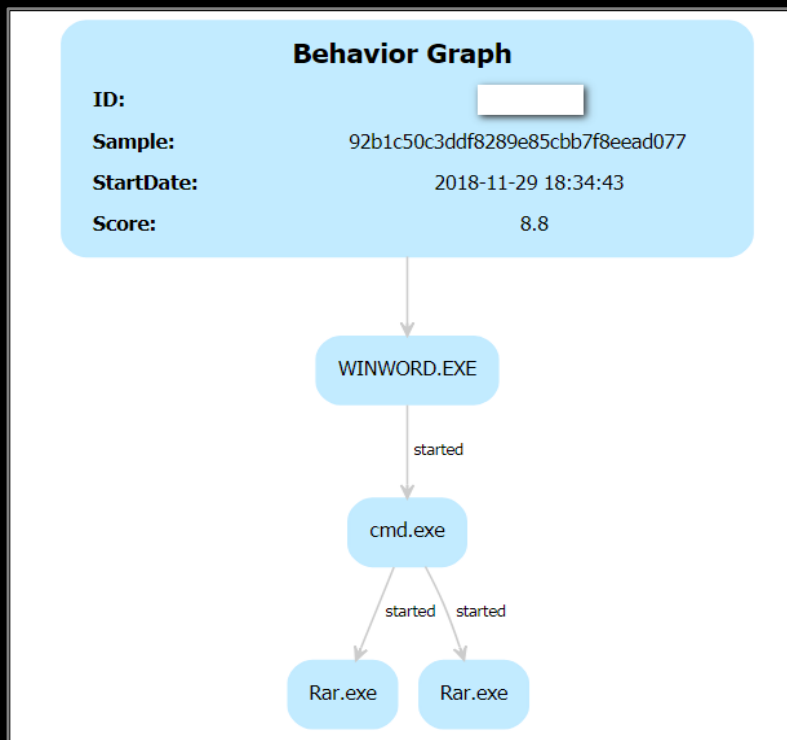
```
class_6
1 package
2 {
3     import avm2.intrinsics.memory.li8;
4     public class class_6
5     {
6         {
7             {
8                 li8(123456);
9             }
10        }
11        public function class_6()
12        {
13            super();
14        }
15    }
16 }
17 }
18 }

1 method
2 name null
3 returns null
4 body
5 maxstack 3
6 localcount 2
7 initscopedepth 3
8 maxscopedepth 6
9 try from ofs0000 to ofs0004 target ofs0004
10
11 code
12 try{
13     jump ofs0024
14 }
15 }
16 catch{
17 ofs0004:
18     local_0 = local_449
19     local_449 = local_448
20     local_448 = local_0
21     jump ofs0028
22 }
23 }
24 ofs0024:
25     li8(123456)
26
27 ofs0028:
28     returnvoid
29 }
```

# CVE-2018-15982



- 2018年11月29日
- 2个小时, 2个样本
- TVSDK 中的 UAF 漏洞



- 2018年12月5日, Adobe 再次对我们进行了致谢

## Acknowledgments

Adobe would like to thank the following individuals and organizations for reporting the relevant issues and for working with Adobe to help protect our customers:

- Chenming Xu and Ed Miles of Gigamon ATR (CVE-2018-15982)
- [Yang Kang \(@dnpushme\) and Jinqun \(@jq0904\) of Qihoo 360 Core Security \(@360CoreSec\) \(CVE-2018-15982\)](#)
- He Zhiqiu, Qu Yifan, Bai Haowen, Zeng Haitao and Gu Liang of 360 Threat Intelligence of 360 Enterprise Security Group (CVE-2018-15982)
- b2ahex (CVE-2018-15982)



# CVE-2018-15982

- 用 HackingTeam 的技巧绕过了 ROP 检测 😊
- 无法躲避 EAF 检测 😞

```
// Virt(ualPro)tect = 74726956 74636574
var vp_addr:uint = this.getFuncAddrByEAT32(0x74726956, 0x74636574, 10, kernel32_addr);

...

this.writeDWORD32(sc_addr + 8 + 0x80 + 0x1c, vp_addr);
this.writeDWORD32(ptbl, sc_addr + 8 + 0x80);
this.writeDWORD32(p + 0x1c, sc_addr);
this.writeDWORD32(p + 0x20, vec_uint.length * 4);
var args:Array = new Array(0x41);
Payload.call.apply(null, args); // Call VirtualProtect to bypass DEP
```

# 其他收获

- 1 Word CVE 🍷
- 1 PowerPoint CVE 🍷
- 4 Excel CVE 🍷
- 1 Win32k CVE 🍷

Microsoft Excel Remote Code Execution Vulnerability	CVE-2018-0920	Yangkang (@dnpushme) &Wanglu of Qihoo360 CoreSecurity @360CoreSec Vladislav Stolyarov of Kaspersky Lab
Microsoft PowerPoint Remote Code Execution Vulnerability	CVE-2018-8376	yangkang(@dnpushme) & Jinquan(@jq0904) & Wanglu of Qihoo360 CoreSecurity(@360CoreSec)
Microsoft Excel Remote Code Execution Vulnerability	CVE-2018-8379	Jinquan(@jq0904) of Qihoo360 CoreSecurity(@360CoreSec) Yangkang(@dnpushme) of Qihoo360 CoreSecurity(@360CoreSec)
Microsoft Word Remote Code Execution Vulnerability	CVE-2018-8539	Yangkang of 360CoreSec Jinquan of 360CoreSec
Microsoft Excel Information Disclosure Vulnerability	CVE-2018-8627	Yangkang(@dnpushme) & Jinquan(@jq0904) of Qihoo360 CoreSecurity(@360CoreSec)
Microsoft Excel Information Disclosure Vulnerability	CVE-2019-0669	Jinquan o 360CoreSec Yangkang of 360CoreSec
Windows GDI Elevation of Privilege Vulnerability	CVE-2018-0817	HongZhenhao Li Qi(@leeqwind) of Qihoo 360

# 总结

- 从1到N易，从0到1难
- 了解对手，反思自己，战胜对手
- 永远在路上



# 致谢



- 感谢 360 高级威胁团队的所有小伙伴
- 感谢 @programmeboy, @guhe120, @binjo, @Unit42\_Intel
- 特别感谢 @HaifeiLi 和他关于 Office 安全的分享

# BLUEHAT

## SHANGHAI 2019

### 大海捞针：使用沙箱捕获多个零日漏洞

李琦 金权

[liqi3-s@360.cn](mailto:liqi3-s@360.cn) [jinquan@360.cn](mailto:jinquan@360.cn)

[@leeqwind](#) [@jq0904](#)